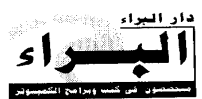
تعلم بدون تعقيد ..

PHP

اللغة المجانية الأولى لبرمجة الويب



المركز الرئيسي : 11 شارع د/محمد رأفت – محطة الرمل – الإسكندرية
تليفون وفاكس : 4838326 (03)(2+)
موبايل : 0101634294 (2+) - 0123357844 (2+)
Email: info@egyptbooks.net
URL: www.egyptbooks.net

اسحاق بخيت
(embekheet@yahoo.com)



وَقُلِ اعْمَلُوا فَسَيَرَى اللَّهُ عَمَلَكُمْ وَرَسُولُهُ وَالْمُؤْمِنُونَ



طبعة 2005

رقم الإيداع

2005/1898

ISBN

977-17-1956-4

الفصل الاول

ما قبل البداية

مقدمة

لغة PHP او (Hypertext Preprocessor) وتم تصميم هذه اللغة فى البداية بواسطة شخص اسمه راسموس ليدروف وكانت يوما جزء من لغة Perl حتى تم تحويلها الى لغة مستقلة و تنقسم الان الى مكتبتين هما Zend من تطوير شركة Zend و مكتبة PHP ويمكنك زيارة الموقع الرسمى للغة (www.php.net) سيمكنك تحميل المكتبتين معا وكود هذه اللغة مفتوح المصدر بعكس الكثير من لغات البرمجة الشائعة كما تتميز بالكثير من الخصائص التى جعلتها من الركائز الاساسية لتطوير برامج ويب امنة ومستقلة وفيما يلى اهم هذه المميزات:

- هذه اللغة سهلة التعلم حيث تحتل المرتبة الاولى بالنسبة للغات برمجة الويب من حيث قصر منحنى التعلم المطلوب و تشبه الى حد كبير من حيث قواعد اللغة الى لغة C++ العريقة ولكن بدون التعقيدات الموجودة بها
- لكى يتم تنفيذ برنامج ويب يجب استضافته على سيرفر معين ومن اشهر السيرفرات التى يمكن ان تعمل عليها البرامج المكتوبة بهذه اللغة هو سيرفر لينكس (Linux) الذى يعتبر نظام تشغيل مجانى تقريبا ورخيص جدا ويتم استخدام برنامج اباتشي المشهور على هذا النظام الذى يستطيع تنظيم واظهار صفحات PHP بسرعة كبيرة كما يمكن ايضا استضافة صفحات هذه اللغة على سيرفرات ويندوز مع البرنامج IIS الذى يأتى بدء من اصداراة ويندوز 2000 وسنتعلم فيما بعد كيفية تركيب البرنامج على النظام ويندوز اكس بى كنظام قياسى لمعظم المطورين.

- تتميز هذه اللغة بوجود مكتبة قوية تحتوى على الكثير من الأدوات الاساسية اللازمة لانتاج برنامج قوى و سريع و امن ومكتبات دوال للتعامل مع لغة وملفات XML و ارسال واستقبال الملفات عن طريق بروتوكولات FTP ويمكنها ايضا الاتصال بجميع انواع قواعد البيانات المشهورة بدء من اكسيس وحتى اوراكل و لكن تركيز استخدام هذه اللغة يكون مع قاعدة البيانات القوية MySQL التى تتميز هى الاخرى بانها مجانية و مفتوحة المصدر للتطوير .
- الكود المحمول (Portable Code) وهو مصطلح شائع يدل على امكانية كتابة الكود مرة واحدة للبرنامج و تنفيذه على انظمة السيرفرات المختلفة سواء ويندوز او لينيكس او يونيكس وهذا يسهل كثيرا حياة المبرمج و يعفيه من قضاء العديد من الساعات الاضافية لمجرد عمل تعديلات .
- فى وقتنا الحالى تثير المشكلة الامنية على الشبكة العالمية اهمية قصوى نظرا لاعتماد الكثير من الانظمة المالية كالبنوك و المتاجر الالكترونية على هذه الشبكة فيصبح تأمين تعامل المستخدم مع البائع او مقدم الخدمة امرا حتميا و تأتى هنا لغة PHP لتزودك بالكثير من الادوات التى تستطيع من خلالها تأمين اتصال المستخدم وتحديد عدد الاتصالات المسموح بها لقاعدة البيانات فى نفس الوقت حيث ان دخول العديد من المخربين على الموقع من جميع انحاء العالم فى نفس الوقت يؤدى لتعطل السيرفر و بالتالى الموقع و هى من الطرق المعروفة لمنظمات الهاكرز العالمية لاختراق و تعطيل المواقع .

- هذه اللغة هى لغة غير محدودة اى لا تنتجها شركة معينة او تقتصر حقوق النشر على مؤسسة بذاتها و لكن يتم تطوير هذه اللغة من جميع المطورين على مستوى العالم فيمكنك ان تقوم بالتعديل مباشرة فى كود اللغة و تزويدها بخصائص معاينة تناسبك او البحث عما تحتاج اليه من الموارد الغير محدودة لهذه اللغة فحقا هذه اللغة لا يمكن ان تموت يوما.

وهذه هى اهم اسباب رخص تكلفة استضافة صفحات بى اتش بى و اقبال الكثير على استخدامها .

و فيما يلى سنبدء بالاعدادت اللازمة للبدء فى البرمجة باستخدام هذه اللغة الممتعة .

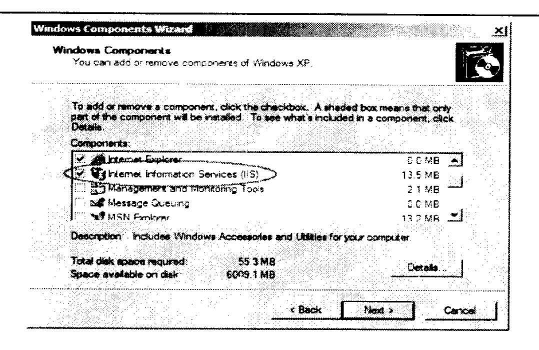
تركيب برنامج IIS

ستحتاج لتثبيت هذا البرنامج الى القرص المدمج لنظام ويندوز اكس بى بروفيشونال و يمكنك التأكد من تنزيل هذا البرنامج من قبل ام لا عن طريق كتابة العنوان (http://localhost) فى سطر العنوان ببرنامج انترنت اكسبلورر فأذا ظهرت صفحتين واحدة لها العنوان Microsoft Windwose XP Professional والاخرى بها ملفات المساعدة الخاصة ببرنامج IIS 5.1 فانك لا تحتاج الى اتباع خطوات هذا القسم و يمكنك الانتقال مباشرة الى القسم الذى يليه .

خطوات تثبيت IIS

1- قم بفتح لوحة التحكم Control Panel من قائمة Start ثم اختار ايقونة Add or Remove Programs ثم اضغط DClick عليها واختار Click المفتاح Add/Remove Windows Components

2- ستهر نافذة تحتوى على جميع برامج الويندوز الافتراضية قم بالتأكد من تفعيل الاختيار Internet Information Services ثم اضغط Next حتى يقوم الويندوز بتثبيت ملفات البرنامج وقد تحتاج الى وجود اسطوانة الويندوز فى مشغل الاسطوانات .



3- يمكنك الان التأكد من تثبيت هذا البرنامج بكتابة localhost فى برنامج انترنت اكسبلورر IE و ستجد الدليل الاساسى Inetpub فى المشغل الافتراضى للنظام فاذا قمت بتنزيل الويندوز على المشغل :C فستجد مسار هذا الدليل C:\Inetpub وهذا هو الدليل الخاص ببرنامج IIS و ستجد بداخله الدليل wwwroot الذى يوجد به محتويات الصفحة التى ظهرت لك بعد كتابة localhost .

ملحوظة: كلمة localhost او رقم البرتوكول 127.0.0.1 يعبران عن الجهاز المحلى بدون الاتصال بالانترنت فلكل جهاز يعمل على الشبكة العالمية رقم بروتوكول منفرد و يعبر جزء من الرقم عن البلد الذى يوجد به الجهاز فمثلا الرقم 192.168.20.5 يعبر عن جهاز فى مصر مثلا .

حتى يمكنك كتابة برامج ويب بلغة PHP يجب تركيب مترجم اللغة وفيما يلى خطوات تنفيذ ذلك :

يجب اولا تنزيل احدث اصدار للمترجم وحاليا وقت تأليف هذا الكتاب كان الاصدارة 5.0.2 هى الاحدث و يمكن تنزيلها مباشرة من الوصلة الاتية:

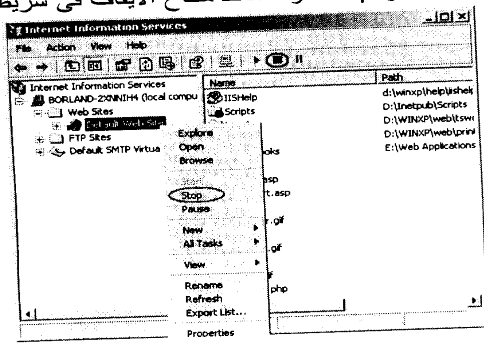
http://fr.php.net/distributions/php-5.0.2-installer.exe

يتم تنزيل الاصدارة رقم 5.0.2 من سيرفر فى فرنسا فاذا ظهرت نسخة احدث يفضل تنزيلها من الموقع الرئيسي www.php.net واختيار النسخة التى تريدها .

بعد تنزيل البرنامج سيكون فى ملف واحد بالامتداد exe وقبل تنزيله يفضل ايقاف عمل برنامج IIS ويتم ذلك عن طريق الخطوات الاتية :

1- قم بفتح ايقونة Administrative Tools من نافذة Control Panel ثم اختار DClick ايقونة Internet Information Services

2- هذه النافذة مقسمة الى جزئين الايسر كشجرة للمواقع و الايمن هو مستعرض لملفات او صفحات هذه المواقع قم الان باختيار Deafault Web Site و اضغط RClick و اختار Stop او اضغط مفتاح الايقاف فى شريط الأدوات .



- قم الان بتشغيل الملف الذى قمت بتنزيله و بعد التنزيل قم بفتح نافذة IIS و قم باعادة تشغيلة عن طريق القائمة المختصرة على Default Web Site و اختيار Start او الضغط على مفتاح Start من شريط الادوات .

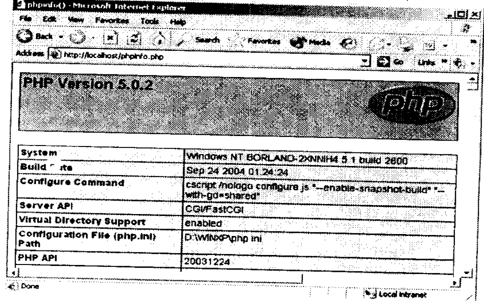
- لتتأكد من عمل المترجم بطريقة صحيحة قم بانشاء ملف نصى جديد عن طريق برنامج NotPad ثم قم بكتابة الكود التالى داخله :

```
<?
 echo phpinfo();
?>
```

وقم بحفظ الملف بالاسم phpinfo.php داخل الدليل c:\inetpub\wwwroot ثم قم بفتح IE وقم بكتابة العنوان التالى :

http://localhost/phpinfo.php

ستجد نافذة المستعرض تظهر صفحة تحتوى على معلومات عن النسخة المثبته للمترجم PHP مثل رقم الاصدار وغيرها :



قمنا حتى الان بتجهيز البرنامج المضيف للصفحات وهو IIS و المترجم PHP و كخطوة اخيرة يجب ان نقوم بتثبيت قاعدة البيانات MySQL التى يمكن عن طريقها انشاء موقع ويب ديناميكى ويمكن تنزيل MySQL مجانا من الموقع www.mysql.com واختيار احدث اصدار .

تركيب ملف PHP:

– الملف الذى يحتوى على كود بلغة php هو مجرد ملف نصى عادى جدا يمكن انشاؤه عن طريق برنامج NotPad البسيط اوالبرامج المختلفة الاخرى مثل Dreamweaver هذا بجانب العديد من برامج تحرير لغة php المجانية التى يمكن الوصول اليها عن طريق الموقع www.download.com .

– اذا قمت بفتح صفحة مكتوبة بلغة php ستجد انها تتكون من جزئين جزء به لغة HTML القياسية و الاخر بلغة php وحتى يمكن كتابة كود بهذه اللغة يجب اخبار المترجم اولا و يتم ذلك بحصر كود php بين علامتين خاصتين و فيما اهم هذه العلامات .

1- كما رأينا يمكن وضع الكود بين العلامتين ?> و <? مثال:

```
<?
 echo "Hello World..."
?>
```

اذا قمت بتنفيذ هذه الصفحة ستجد عبارة "Hello World..." تم طبعها فى الصفحة الرئيسية للمستعرض .

2- يمكن ايضا استخدام عبارة التعريف القياسية php كالتالى:

```
<?php
  echo "Hello World... "
?>
```

3- يمكنك استخدام عبارة تحديد اللغة script language كالتالى :

```
<script language = "php">
 echo "Hello World... "
/script>
```

4- عن طريق علامة السكريبت القياسية %> و هى تستخدم ايضا مع كل من لغة VBScript و لغة JavaScript لصفحات ASP .

```
<%
 echo "Hello World... "
%>
```

كيف يتم تنفيذ صفحات php ؟

هناك طريقتين لتنفيذ هذه الصفحات الاولى هى نسخ الصفحات المراد استعراضها تحت الدليل c:\inetpub\wwwroot و فى هذه الحالة يتم الوصول الى الصفحات المراد استعراضها عن طريق الوصلة الاتية

http://localhost/page1.php

حيث page1.php هى الصفحة المراد تنفيذها

و الطريقة الاخرى هى بانشاء دليل تخيلى للدليل الذى يوجد به صفحات php المراد استعراضها و يتم ذلك عن طريق الخطوات الاتية:

– قم بفتح برنامج IIS و اضغط RClick على Default Web Site و اختار New | Virtual Directory سيظهر المرشد الخاص بانشاء الدليل .

– اضغط Next حتى تنتقل الى الصفحة التالية و قم بكتابة اسم Alias و هو الاسم التخيلى للدليل و ليكن "myphp" .

– اضغط Next للانتقال الى الصفحة التالية واختار الدليل الفعلى الذى توجد به صفحات php وليكن "e:\php\prog1" ثم اضغط Next .

– ستظهر صفحة بها الصلاحيات التى تريد منحها للموقع مثل القراءة فقط او تنفيذ برامج CGI قم بقبول الاختيارات الافتراضية و اضغط Next ثم Finish .

– لكى تقوم باستعراض الصفحة يجب كتابة العنوان التالى فى IE

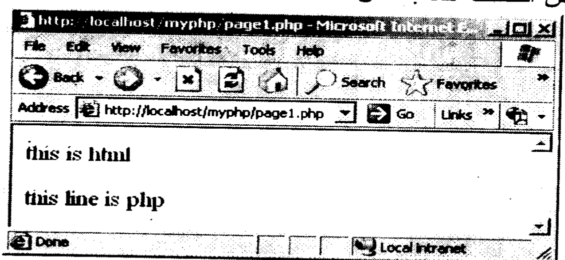
http://localhost/myphp/page1.php

مثال:

يمكنك التفريق بين مخرجات لغة HTML و لغة PHP من الكود الاتى:

```
<P> This is html </P>
<?
 echo "This line is php"
?>
```

سيتم عرض الصفحة كما بالشكل:

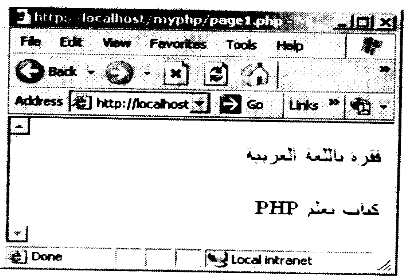


وسنقوم فيما بعد بشرح قواعد لغة php

ملحوظة: اذا اردت اظهار نص باللغة العربية يجب استخدام العروة dir الخاصة بلغة HTML لاظهار اللغة من اليمين الى اليسار و فيما يلى كود يؤدى ذلك:

```
<html dir="rtl">
<p>فقرة باللغة العربية</p>
<?
echo "كتاب تعلم PHP"
?>
```

تكون النتيجة كما بالشكل:



لاحظ 1: تحرك شريط التمرير الايمن الى اليسار ليناسب الاعداد الجديد
لاحظ 2: عند كتابة العنوان فى سطر العنوان الخاص ببرنامج IE يتم تنفيذ عملية تسمى Request او طلب من السرفر و يقوم IIS بتنظيم ذلك و الاستجابة للطلب حتى يتم عرض الصفحة للمستخدم بعملية تسمى Response و الفرق بين الوقتين هو وقت الاستجابة و يتوقف عليه عوامل كثيرة جدا مثل سرعة السرفر و عدد المواقع به و لكن العامل الوحيد الذى نستطيع التحكم به لتقليل هذا الوقت هو حجم الصفحة من رسومات او افلام متحركة فيجب مراعاة عامل سرعة عرض الصفحة بحيث لا تأخذ عملية تحميل الصفحة لمستخدم النت عن طريق Dial-up الى اكثر من 30 ثانية والا سيمل الزائر و يتجه الى موقع اخر وفى هذه العملية يتم ترجمة اى سكريبت قمت بكتابته الى صفحة صالحة للعرض فاذا حاولت ان تشاهد مثلا الكود الخاص بك من صفحة موجودة على النت لن تشاهد الا نتيجة هذه المعالجة وكود HTML فقط .

يتم اثناء هذه العملية استخدام برتوكولات خاصة حتى يتم تبادل المعلومات بين السرفر و المتصفح و فيما يلى شرح لهذه البرتوكولات:

برتوكول TCP/IP:

هذا البرتوكول اساسى جدا للتحكم فى عملية نقل البيانات خلال شبكة النت العالمية ولكى تفهم طريقة عمله تخيل انه مثل ساعى البريد ولكن فى حالتنا يقوم عدة سعاة بتوصيل رسالة واحدة و فى لغة الانترنت تسمى حزمة او Packet بحيث تنقسم هذه الحزمة الى جزئين الاول يحمل البيانات المراد نقلها و الثانى يحمل عنوان الوصول الذى يجب توصيل البيانات اليه و يقوم هذا البرتوكول

كمنظم للمرور من الراسل و الى المرسل اليه اما برتوكول http فيقوم بعنونة الحزمة .

فاذا تم فقد احد الحزم فلن تنتقل الرسالة بطريقه صحيحة الى المرسل اليه لذلك يتم اخراج حزمة خاصة تسد الفقد الحادث اثناء نقل البيانات ويحدث هذا بسبب سؤ الخط او مشكلة فى جهاز الراسل او جهاز المرسل اليه و يتعرف البرتوكول على وجود حزمة ناقصة عن طريق عدد الحزم و حجم البيانات المرسله فيقوم بعملية تأكد من وصول كل الحزم الى طالب الخدمة و يتم كل هذا فى ثوان او اقل حسب نوع الاتصال .

يتعرف برتوكول النقل على الجهاز الذى يطلب الصفحة عن طريق رقم منفرد يسمى IP وهو عبارة عن اربع مجموعات للارقام تحدد البلد والمدينة للجهاز المتصل بالنت فيتم وصول الحزم الى هذا الجهاز و قد يتم توليد هذا الرقم عشوائيا او يكون ثابتا حسب نوع الاتصال بالنت .

برتوكول HTTP:

كما ذكرنا ان عملية طلب الصفحة تسمى Request و يتم فيها تحديد الصفحة المراد اظهارها وتسمى بالصطلح URL او (Uniform Resource Locator)

و يتم عند طلب اجراء عملية بحث مثلا على الانترنت بارسال معلومات اضافية بجانب الصفحة المراد اظهارها هذه المعلومات تحدد معاملات البحث فاذا كنت تبحث عن كتاب مثلا بجزء من اسم الكتاب و المؤلف فيتم ارسال هذه المعلومات كجزء من عنوان (URL) الى صفحة البحث للتعامل معها واظهار النتيجة التى يريدها المستخدم فى عملية الاستجابة Response .

و تنقسم عناوين http ال ثلاثة اقسام حسب نوع العنوان اذا كان طلب او استجابة (Request / response line - HTTP header - HTTP body) و فيما يلى شرح مفصل لهذه الاقسام حسب نوع العنوان:

- **حالة الطلب Request:**

– يحتوى السطر الاول من عنوان http على الامر المستخدم او method و المسار من الموقع الى الصفحة المطلوبة و رقم اصدارة http .

– الجزء الثانى يسمى header و يحتوى على تفاصيل عن العميل مثل نوع المتصفح و رقم الاصدار والوقت و التاريخ و هذا الجزء يمكن تقسيمه من حيث البيانات التى يحملها الى ثلاثة انواع: بيانات عامة لاتخص جهاز المستخدم – بيانات خاصة تحتوى على معلومات عن البيانات المرسله – بيانات مطلوبة تحتوى على اعدادات العميل .

– الجزء الاخير يسمى body و عند استخدام امر post للعنوان يتم استخدام هذا الجزء لارسال بيانات الى السرفر .

- **حالة الاستجابه Response:**

– يحتوى الجزء الاول على سطر للاستجابة و هو ينقسم الى رقم اصدار http وناتج عملية الطلب اذا كانت ناجحة ام فاشلة .

– الجزء الثانى يشبه حالة الطلب و ينقسم الى معلومات عامة ومعلومات خاصة ومعلومات الاستجابه التى تحتوى معلومات عن السيرفر الذي قام بإرسال الرد وكيفية التعامل ومعالجة الرد .

– الجزء الخير body و يحتوى على كود html فى حالة تنفيذ الرد بنجاح و يقوم المستعرض بترجمة الكود الى الصفحة النهائية .

اجراء ترجمة كود PHP:

تعتمد لغة php على تنفيذ كودها من جهة السرفر و ليس محليا فى جهاز المستعرض و يقوم السرفر بتنفيذ عمليتن على هذا الكود هما:

– عملية الفحص و تسمى Parsing و فيها يتأكد المترجم من صحة الاوامر المكتوبة لغويا وليس منطقيا .

– عملية تنفيذ وتسمى Executing وهى التى تنتج صفحة html النهائية .

الفصل الثانى

قواعد لغة PHP

قواعد لغة PHP:

فيما يلى سنقوم بشرح لاهم قواعد لغة PHP و كيفية كتابة الاوامر .

التعليقات:

هى مجرد سطور تشرح المغزى من جزء معين من الكود و تظهر فائدتها عند كتابة برنامج كبير وتعديله بعد مرور سنوات فيصبح من العسير جدا على المطور تذكر فائدة السطور التى كتبها او المخرجات التى ستنتج عنها لذلك يتم ادراج التعليقات كشرح للمطور ليس اكثر و لا يقوم المعالج الخاص باللغة بترجمة هذه التعليقات او الالتفات اليها .

والسؤال الان كيف يعرف المترجم سطور التعليقات من سطور الكود ؟

يتم ذلك عن طريق علامات خاصة و توجد طريقتين الاولى يمكن فيها ادراج تعليق بعد العلامة // حتى وان كان يسبقها سطر كود عادى

مثال:

```
<?
echo "hi..."; //هذا سطر تعليق
?>
```

والطريقة الثانية تستخدم لادراج التعليق على اكثر من سطر عن طريق علامة */ هكذا :

```
<?
 /* سطر تعليق 1
 سطر تعليق 2 */
  echo "hi...";
?>
```

المتغيرات:

هى من اساسيات اى لغة و تستخدم لتخزين قيمة معينة فى ذاكرة الكمبيوتر و يتم تعريف المتغير عن طريق علامة الدولار "$" بالصيغة الاتية:

$اسم_المتغير=قيمة المتغير;

نلاحظ من الصيغة السابقة انه يتم اختيار الاسم الذى نريده للمتغير (يجب ان يكون بالانجليزية) ووضع العلامة $ قبله ثم نكتب علامة التساوى التى تقوم بتخزين القيمة فى المتغيرو اخيرا الفاصلة المنقوطة اخر اى سطر من سطور لغة PHP .

قيمة المتغير قد تكون نصية او عددية فى المثال التالى سنقوم بتخزين قيمة نصية فى متغير اسمه name:

```
$name="بخيت";
```

لغة PHP حساسة لحالة الاحرف فمثلا لا يمكن تخصيص قيمة للمتغير name ثم فحص قيمتها بالنداء على المتغير Name فى هذه الحالة يكون الاسمين متغيرين منفصلين تماما .

مثال:

```
$name="رامى";
$Name="ايهاب";
echo $name;
```

هل تستطيع توقع اى الاسمين سيتم طباعته ؟

وكقاعدة عامة لتسهيل عملية تطوير البرامج و لكى تستطيع تذكر الهدف من المتغير بسهولة يجب ان يعبر اسم المتغير عن وظيفته ولايسمح بمسافات خالية فى اسم المتغير بل يجب ان يكون متصل لذلك تستطيع استخدام علامة "_" بين الكلمات فمثلا اسم متغير لتخزين درجة الحرارة يكون نموذجيا بالاسم $heat_degree وهكذا .

نلاحظ من الامثلة السابقة اننا قمنا بتخزين قيم نصية او حرفية ويتم التعبير عنها بين علامتى التنصيص "" و هى قيم لا يمكن اجراء عمليات حسابية عليها اما اذا اردنا تخزين قيم عددية يمكن اجراء عمليات حسابيه لها يجب ان نتعرف اولا على انواع البيانات الممكن استخدامها فى لغة PHP .

انواع البيانات:

1- بيانات حرفية:

يعتبر معالج لغة PHP اى قيمة موضوعة بين علامتى التنصيص مفردة او مزدوجة على انها قيمة نصية وفيما يلى امثله لذلك:

```
$var1="text";
$var2='some string...';
```

واذا اردت ادراج العلامة ' فيجب كتابة النص بين العلامتين "" اما اذا اردت كتابة العلامة" داخل النص فيجب كتابتها بعد العلامة \ كما يلى:

```
$str_var="my name is \"bekheet\""
```

اذا اردت طباعة مسار ملف مثلا بحيث يتم وضع العلامتين \\ بجانب بعضهم فى هذه الحالة يجب كتابة القيمة كمايلى:

```
$v="c:\\\windwos\\\system32";
```

وتكون نتيجة طباعة هذا السطر هى

```
c:\\windows\\system32
```

وفى حالة اذا قمنا بتخزين رقم الشارع فى متغير و اسم المدينة فى متغير اخر فلكى نستطيع طباعة العنوان كاملا يجب ان نقوم بربط المتغيرين فى متغير جديد كما يلى:

```
$v_st="306 st,";
$v_city="Alexandria";
$full_Address=v_st.' '.v_city;
```

وتكون النتيجة هى السطر:

```
306 st, Alexandria
```

لاحظ هنا اننا قمنا بالربط عن طريق علامة النقطة (.) و قمنا باضافة مسافة خالية حتى لايكون الكلام ملتصق ببعض .

ملحوظة: يمكنك عند اختبار الامثلة فتح نسخة واحدة من IE و الضغط فقط على مفتاح F5 فى كل مرة تقوم فيها بتعديل الكود لتشاهد نتيجة التعديل .

2- البيانات العددية:

وهى نوعان الاعداد الصحيحة و الاعداد العشرية او ذو الدقة المضاعفة و يتم التعبير عن النوع الاول بكتابة الرقم بدون علامات تنصيص و النوع الثانى يحتوى بالطبع على كسر عشرى ولا تعطى اهتمام كبير للنوعين لان المعالج

يقوم بالتحويل بين النوعين حسب نوع القيمة او ناتج العملية الحسابية لنفس المتغير وفيما يلى مثال لهذا النوع من المتغيرات :

```
$n1=3; // متغير صحيح
$n2=5.89;// متغير عشرى
$n1= $n1/$n2; // المتغيرالان أصبح عشرى
echo $n1;
```

وتكون النتيجة هى طباعة القيمة 0.509337860781

تأتى هنا اهمية معرفة العلامات الخاصة بالعمليات الحسابية وهى "+" لعملية الجمع و "-" لعملية الطرح و "/" لعملية القسمة و "*" لعملية الضرب

و هناك قاعدة هامة لاولوية الحساب فمثلا المعادلة (5*2+7) تكون نتيجتها 45 او 17 لمعرفة ذلك اتبع القاعدة التالية:

1- يتم حساب الارقام بداخل الاقواس اولا

2- يتم حساب عملية الضرب او القسمة ايهما اولا من اليسار لليمين

3- يتم حساب عملية الجمع او الطرح ايهما اولا من اليسار لليمين

مما سبق تكون المعادلة السابقة نتيجتها 17 فاذا اردت اجراء عملية الجمع اولا فيجب ان تضعها بين اقواس هكذا 5*(2+7) فتكون النتيجة 45

هذه اللغة تشبه الى حد كبير لغة C++ و يظهر ذلك عند عملية زيادة قيمة المتغير بمقدار واحد او باضافة المتغير الى نفسه مرتين كما يلى:

$i = $i + 1; //الوضع الاعتيادى للزيادة
$i++; //زيادة بمقدار واحد على طريقة لغة السى
$i+ = 2; // 2 زيادة المتغير بمقدار

$i = $i + $i; //اضافة المتغير الى نفسه بالطريقه العادية
$i+=$i; // اضافة المتغير الى نفسه على طريقة السى

ولا فرق بين الطريقة العادية و طريقة السى فى النتيجة ولكن طريقة السى فقط تجعلك تبدو كمحترف اكثر

متغيرات النظام:

هى متغيرات اسمها محجوز لدى معالج اللغة بحيث يقوم باستبدال المتغير بقيمة معينة بمجرد رؤية هذا المتغير.

الثوابت:

من مكونات اى لغة برمجة وتستخدم لتخزين قيم ثابتة بحيث لا يمكن اجراء عمليات حسابية عليها وتغيير قيمتها وفائدتها للتأكد من ثبات قيمة معينة اثناء تنفيذ برنامج كبير بحيث لا يتم تغيير هذه القيمة على سبيل الخطأ .

ويتم اداء ذلك عن طريق العبارة define التى تأخذ معاملين الاول هو اسم الثابت و الثانى هو قيمة هذا الثابت و يجب ان يكون كلاهما بين علامتى التنصيص كما يلى :

```
<?
define("Pi","3.14");
echo "Pi value is... ".' '.Pi;
?>
```

وهناك ايضا ثوابت محجوزة لدى النظام مثل الثابت PHP_OS الذى يقوم بتحديد نوع نظام التشغيل المستخدم (السرفر)

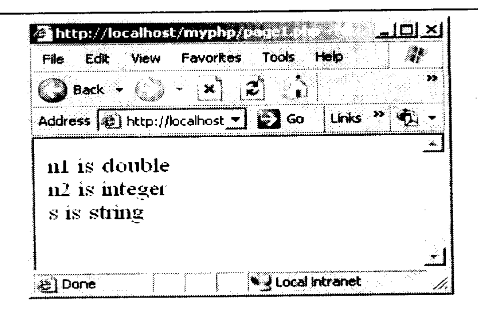
```
echo PHP_OS;
```

التحويل بين انواع البيانات:

فى بعض الاحيان يكون من الهام معرفة نوع المتغير و يتم ذلك بالنداء على دالة بسيطة تسمى gettype() و يمكن اختبارها كالتالى:

```
<?
$n1=3.5;
$n2=3;
$s="word";
echo 'n1 is '.gettype($n1).'<br>';
echo 'n2 is '.gettype($n2).'<br>';
echo 's is '.gettype($s).'<br>';
?>
```

وتكون النتيجة كالتالى :



اما اذا اردنا تحويل متغير معين من نوع الى اخر يمكن اداء ذلك عن طريق الدالة settype ويتم ذلك عن طريق معاملين الاول لاسم المتغير و الثانى لنوع المتغير الجديد كما يتضح من المثال التالى:

```
$n1= 10; //integer
settype($n1, "string");
echo gettype($n1);
```

الدالة isset():

عند كتابة برنامج كبير (اكبر من 1000 سطر) يكون من المستحيل معرفة اذا كان متغير معين قد تم استخدامه من قبل ام لا فاذا قمت مثلا بتخزين قيمة الدرجة فى متغير يسمى degree حتى تقوم فى اخر البرنامج بعرض هذه القيمة ونسيت وانت تكتب البرنامج وقمت باستخدام نفس المتغير لتخزين قيمة اخرى فستكون نتيجة الكود الذى قمت بكتابته غير صحيحه لذلك يجب اختبار المتغير اذا كنت تشك فى وجوده من قبل هكذا:

```
echo isset($n);
```

فاذا لم يكن المتغير موجود من قبل فان الدالةلا تعطى قيمه (null) واذا كان موجود فانها تعطى القيمة 1 ويمكن اختبار قيمتها عن طريق عباراة الشرط IF كما سنرى فيما بعد .

الدالة unset():

وهى تحذف المتغير تماما من الذاكرة وتجعل هذا الجزء من الذاكرة صالح للاستخدام مرة اخرى يجب التأكد من عدم الحاجة الى المتغير او النداء عليه قبل استخدام هذه الدالة .

مثال:

```
unset($n);
```

الدالة empty():

وهذه الدلة تختبر قيمة المتغير اذا كانت خالية "" او صفر او متغير غير موجود فانها ترجع القيمة 1 و لا تقوم بارجاع اى شئ اخر خلاف ذلك .

مثال

```
$n=0;
echo empty($n).'<br>';
$n=2
echo empty($n);
```

و النتيجة لن تقوم الدالة فى السطر الاول بارجاع اى قيمة وفى السطر الثانى تقوم بارجاع القيمة 1

دوال الوقت و التاريخ:

تحتوى لغة PHP على العديد من الدوال المفيدة التى تقوم باختصار الوقت لنا ويجب تذكر استخدام كل دالة حتى تستطيع استخدامها فى الوقت المناسب وفيما يلى كيفية استخدام دوال الزمن:

دالة (gmdate([format]):

يقوم هذه الدالة بارجاع قيمة الوقت و الزمن الحاليين بالتشكيل المحدد للمعامل [format] كما سنرى فأذا اردت طباعة الشهر الحالى فقط فى الصفحة فأكتب الكود :

```
<?
Echo gmdate ("m");
Echo "\t"; //مسافة خالية
Echo gmdate ("M");
?>
```

تلاحظ برنامج IE يقوم بعرض الشهر الحالى للسطر الاول كقيمة عددية ثم يقوم بطباعة مسافة خالية ثم اسم الشهر الحالى مختصر كقيمة نصية

12 Dec

من هنا نلاحظ ان استخدام الحروف الكبيرة ليست مثل استخدام الحروف الصغيرة :

مثال:

اذا اردت عرض التاريخ بالشكل يوم/شهر/سنة فقم باستخدام الدالة كالتالى:

```
echo gmdate"("d/m/Y")
```

فتكون النتيجة هى طباعة التاريخ 30/12/2004 (مثلا) فنلاحظ من هنا ان الحرف d يرمز لليوم و الحرف m يرمز للشهر و الحرف y يرمز للسنة .

أما بالنسبة للوقت فنفس القاعدة حرف h يرمز للساعة و حرف i يرمز للدقائق و حرف s يرمز للثوانى .

مثال:

الكود الاتى يقوم بطبع النتيجة: 12:24:53 pm

```
echo Gmdate("h:i:s a");
```

الفصل الثالث

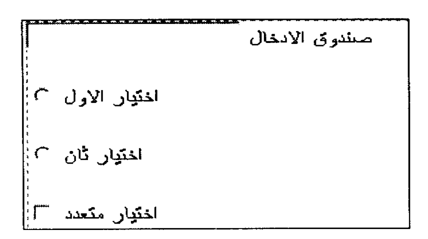
النماذج

استخدام النماذج Form :

تستخدم النماذج لاستقبال قيم او مدخلات من المستخدم او لعرض قيم مخزنة بقاعدة بيانات كنتيجة لعملية بحث مثلا و اكثر مثال على النماذج هو الصفحة التى تجدها دائما عند بدء الاشتراك فى خدمة يقدمها موقع معين مثل مواقع البريد الالكترونى فيطلب منك الموقع تسجيل اسمك و سنك و غيره من البيانات و يقوم بتخزين هذه البيانات تلقائيا .

و يوجد دائما فى نهاية الصفحة التى تقوم بالادخال فيها مفتاح يسمى submit و الغرض منه هو ارسال هذه البيانات لمعالجتها

وتتم عملية الادخال عن طريق ادوات قياسية مثل صناديق الادخال (Text Box) او مفاتيح الاختيارات (Radio buttons and check box) و فيما يلى شكل لكل منهم :



والفرق بين Radio و Check هو ان الاول يمكن تنفيذ اختيار واحد فقط و الثانى يعطى للمستخدم عدة اختيارات و سنعرف فيما بعد كيف يمكن ادراج هذه الادوات.

هذه الادوات يجب ان توضع داخل النموذج و يتم الاعلان عن النموذج بالعبارتين

```
<Form>

</Form>
```

من اهم خصائص النموذج العبارتين Action و Method و فيما يلى شرح لكل منهم:

الخاصية Action:

تكتب على الصيغة

```
<Form Action="Result_Page.php">

</Form>
```

فتدل الخاصية على الصفحة التى يجب على برنامج السيرفر ان يقوم بالانتقال اليها عندما يقوم المستخدم بالانتهاء من عملية الادخال و يضغط على مفتاح submit و يوجد بهذه الصفحة (Result_Page.php) الكود اللازم لمعالجة البيانات و اظهار رسالة ترحيب (مثلا) للمستخدم عند الاشتراك .

الخاصية Method:

و قد تكون احدى الحالتين :

```
<Form Action="Result_Page.php" Method="Post">
<Form Action="Result_Page.php" Method="Get">
```

وهما طريقتين مختلفتين لارسال البيانات للصفحة (result_page.php)

و فيما يلى الفرق بين الطريقتين:

الوسيلة Get:

تقوم بارسال البيانات التى قام المستخدم بادخالها عن طريق سطر العنوان بحيث يظهر سطر العنوان مضافا اليه المعامل ؟ ثم اسماء المتغيرات و قيمها .

http://localhost/order_page.php?name=value&age=20

و تأتى المسميات name و age من الخاصية name لادوات الادخال فعند وضع Text Box يمكن اعطاؤه الاسم age عند طريق الخاصية name بحيث تستطيع معرفة ان هذا الصندوق خاص بادخال عمر المستخدم .

و لا يمكن كتابة اسماء ادوات الادخال الا باللغة الانجليزية و لكن المدخلات تكون باللغة المراد استخدامها .

و السطر السابق الذى يظهر فى عنوان الصفحة يسمى سطر الاستعلام او query string و يمكن تشفيرة بحيث لا يظهر للمستخدم او الهاكرز القيم المدخلة الى الموقع اذا كانت البيانات سرية مثل بطاقات الائتمان .

و نلاحظ من هنا او الطريقة Get لا تناسب ارسال بيانات حجمها كبير الى صفحة اخرى لذلك يمكن استخدام الطريقة الثانية كما سنرى .

الوسيلة Post:

وهى نفس طريقة Get ولكن تستطيع ارسال بيانات بكمية اكبر ويتم هذا عن طريق ارسال البيانات من خلال اتصال مستقل وتتميز هذه الطريقة بانها اكثر امنا و لكن على نقيض الطريقة Get فأنها اقل سرعة .

لذلك يتم استخدام الوسيلة Get عادة فى محركات البحث وبذلك يستطيع الاحتفاظ بنتيجة البحث المتمثل فى العنوان URL و فتحة مرة اخرى دون اعادة البحث .

أدوات الادخال فى النماذج:

كما ذكرنا يوجد بعض ادوات الادخال القياسية مثل صندوق الكتابة Text Box و مفاتيح الاختيار Radio and Check buutons و يمكنك وضعها فى الصفحة عن طريق كتابة كود قياسى هو .

```
<input type="[نوع اداة التحكم]" name="[أسم الاداة فى الكود]"
value="[القيمة التى تعرضها الاداة]">
```

مما سبق أذا اردت ادراج صندوق نصى فى الصفحة فأكتب الكود:

```
<input type="text" name="textfield" value="اهلا">
```

وتكون النتيجة كما بالشكل :

اهلا

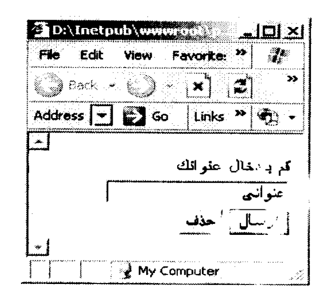
مثال:

سنقوم الان بكتابة ملفين الاول به كود عادى HTML و الاخر به كود PHP يقوم باظهار نتيجة مدخلات المستخدم فى الصفحة الاولى و فيما يلى خطوات اداء ذلك:

– قم بفتح ملف جديد (يمكنك استخدام Notepad او اى محرر اخر مناسب) و امتب فى الملف الاول الكود الاتى:

```
<html dir ="rtl">
<FORM METHOD="GET" ACTION="result.php">
قم بادخال عنوانك
<br>
<INPUT TYPE ="text" NAME = "myaddr" value="عنوانى">
<br>
<INPUT TYPE= submit  VALUE="إرسال">
<INPUT TYPE= reset  VALUE="حذف">
</form>
</html>
```

و الذى تكون نتيجته :



– قم الان بحفظ الصفحة المفتوحة بالاسم page1.htm

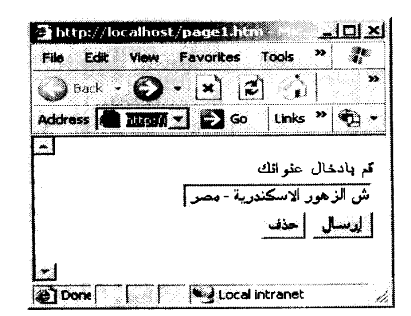
– قم الان بانشاء الصفحة الثانية من المثال و قم بتسميتها **result.php** و اكتب الكود الاتى بداخلها:

```
<?
Echo "عنوانك هو: "." ".$_GET["myaddr"];
?>
```

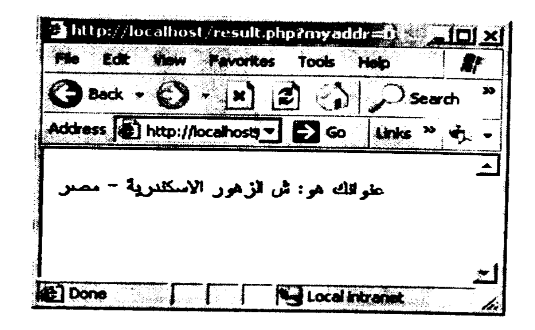
– قم الان بحفظ الملف الثانى بالاسم result.php فى الدليل \:c Inetpub\wwwroot ثم قم بفتح برنامج IE و اكتب العنوان الاتى:

http://localhost/page1.htm

ستظهر أمامك النافذة الاولى قم بكتابة اى قيمة تريدها كما بالشكل:



-بعد الانتهاء من الادخال قم بالضغط على المفتاح ارسال و يجب ان تكون النتيجة كما بالشكل:



ملاحظات على المثال:

- تم ارسال القيمة المدخلة كجزء من العنوان بالشكل:

http://localhost/result.php?myaddr=%D4+%C7%E1%D2%E5%E6%D1+%C7%E1%C7%D3%DF%E4%CF%D1%ED%C9+-+%E3%D5%D1

- فى الصفحة الاولى لم نحتاج الى كتابة لغة PHP لان لغة HTML القياسية توفر ادوات التحكم فى النماذج

- قمنا فى الصفحة الاولى بتحديد نموذج Form والصفحة التى ستتلقى القيم او المتغيرات من الصفحة الحالية

– قم بادراج TextBox و سميناه بالاسم "myaddr" وهو نفسه اسم المتغير الذى سيتم ارساله الى الصفحة result.php

– كما ذكرنا يجب ان يكون لكل نموذج مفتاح واحد على الاقل يقوم بعملية submit او تنفيذ النموذج و هنا قمنا بتعريب المفتاح الى "ارسال" و يمكن ايضا ادراج مفتاح لالغاء القيمة المدخلة الى القيمة الافتراضية كما فعلنا .

– بعد الضغط على المفتاح "ارسال" تم ارسال البيانات الصفحة المحددة فى الخاصية Action و فى الصفحة result.php تم الوصول الى المتغير المرسل عن طريق المصفوفة [""]$_GET التى تحتوى على جميع المتغيرات المرسلة بالطريقة GET و لا يبقى سوى كتابة اسم المتغير بين علامتى التنصيص هكذا ["myaddr"]$_GET ليتم استعراض قيمته .

محرر نصوص:

يمكن ادراج محرر نصى بسيط فى الصفحة اذا كان هناك حاجة لادخال عدة اسطر من النصوص الحرفية و يتم ذلك عن طريق الوسم:

```
<textarea name="" rows=رقم cols=رقم >
</textarea>
```

فيمكن تحديد عدد الصفوف عن طريق الخاصية rows وعدد الاعمدة عن طريق الخاصية cols

مثال:

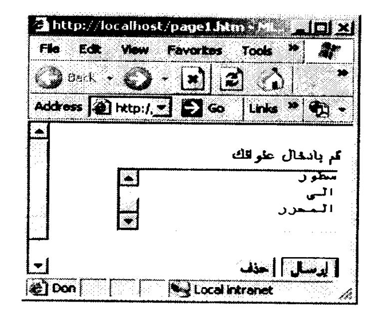
قم بوضع الوسم الاتى:

```
<textarea name="textarea" rows="4"
cols="20"></textarea>
```

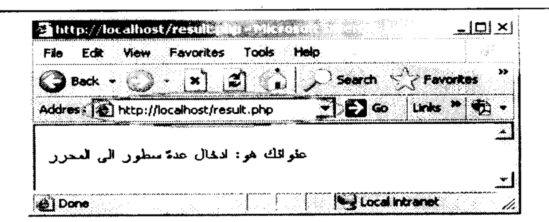
فى الصفحة السابقة page1.htm و عدل الوسيلة GET بالوسيلة POST ثم عدل فى الصفحة result.php بحيث يتم استعراض سطور المحرر عن طريق استبدال السطر $_GET بالسطر الاتى:

```
$_POST["textarea"];
```

والان يجب ان تكون اول نافذة كما بالشكل:



و بعد الضغط على مفتاح الارسال تكون النتيجة:



ملاحظات:

يتضح من هنا الاختلاف بين الطريقة GET و الطريقة POST و لا يظهر فى سطر العنوان قيم المتغيرات .

مربعات الاختيار المتعدد (Check Box):

تظهر فائدتها عندما تريد الاختيار من بين اشياء متعددة مثلا اذا قمت بزيارة موقع للاشتراك فى عدد من المجلات الالكترونية فيتم عرض هذه المجلات فى صورة عدة اختيارات و يقوم المستخدم بالاختيار منها وعلى المطور معرفة اختيارات الزائر وفيما يلى الوسم القياسى لهذه الاداة:

```
<input type="checkbox" name="[name]" value="[value]"
checked>
```

نلاحظ انه تم تحديد نوع الاداة بعد الخاصية type بأنها checkbox و أسم الاداة فى الخاصية name التى يمكن عن طريقه الوصول الى هذه الاداة ثم الخاصية value و هى القمة التى سيتم تخزينها فى متغير الاداة اذا قام المستخدم باختيارها و اخيرا المعامل checked يحدد ما اذا كان صندوق الاختيار مختارا افتراضيا ام لا

مثال:

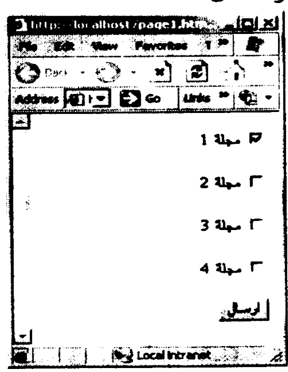
قم بفتح الصفحة السابقة و قم بالغاء المثال السابق أو احتفظ به اذا اردت ثم اكتب الكود الاتى:

```
<html dir="rtl">
</head>

<body>
<form name="form1" method="post"
action="result.php">
  <label>
  <input type="checkbox" name="mag[]" value="m1"
checked>
  مجلة 1</label>
  <p>
    <label>
    <input type="checkbox" name="mag[]" value="m2">
    مجلة 2</label>
  </p>
  <p>    <label></label>
    <label>
    <input type="checkbox" name="mag[]" value="m3">
    مجلة 3</label>
</p>
  <p>
    <label>
    <input type="checkbox" name="mag[]" value="m4">
    مجلة 4</label>
</p>
```

```
  <p>
    <label>
    <input type="submit" name="Submit" value="ارسال">
    </label>
  </p>
</form>
</body>
</html>
```

وتكون نتيجة هذا الكود هو الشكل:



نلاحظ تحميل الصفحة والاختيار الاول فعالا افتراضيا

قم ايضا بتغيير الكود فى الصفحة result.php الى الكود:

```
<?
echo $_POST["mag"][0]."<br>";
echo $_POST["mag"][1]."<br>";
echo $_POST["mag"][2]."<br>";
echo $_POST["mag"][3];
?>
```

قم الان باختيار كل صناديق الاختيار و اضغط **على** المفتاح "ارسال" ستجد فى صفحة result.php انه تم استعراض كل قيم الصناديق و فيما **يلى** كيفية حدوث ذلك:

قمنا **فى الصفحة** الاولى بتسمية كل صناديق **الاختيار** بالاسم **نفسه** وهو "mag[]" **و هذا يعنى** الاعلان عن مصفوفة اى متغير يستطيع ان يحتوى على عدة **قيم فى ان** واحد و يتم الوصول الى كل قيمة عن طريق رقم ترتيبها فى **المصفوفة بحيث تبدء** من الصفر وهذا ما تلاحظة فى صفحة التالية :

توجد **فى الكود السابق** مشكلة وهى اذا لم يقة المستخدم باختيار صندوق فأن رقم **المصفوفة لهذا الصندوق** يصبح غير معرف وبالتالى فمحاولة الوصول اليه تؤدى الى **ظهور خطأ لذلك** يجب اختبار المصفوفة اولا و يتم ذلك عن طريق الدالة isset() **التى قمنا** بشرحها لاحقا و يتم ذلك عن طريق العبارة الشرطية if بحيث يصبح **الكود** السالبق كما يلى:

```
<html dir = "rtl">
<?
if (isset($_POST["mag"][0]))
 echo $_POST["mag"][0]."<br>";

if (isset($_POST["mag"][1]))
 echo $_POST["mag"][1]."<br>";

if (isset($_POST["mag"][2]))
 echo $_POST["mag"][2]."<br>";

if (isset($_POST["mag"][3]))
 echo $_POST["mag"][3];
```

```
?>
</html>
```

بهذه الطريقة يتم اختبار ما اذا كان المتغير موجود اولا ام لا وبنفس الطريقة يمكن التأكد من ادخال قيمة فى خانة معينة قبل معالجة البيانات وهذا امر شائع خصوصا فى النماذج الخاصة بالتسجيل و الشراء فأذا لم يتحقق الشرط من المفروض ان يتم التوجيه الى صفحة التسجيل مرة اخرى بحيث يقوم المستخدم بتصحيح بيناته

أدوات الاختيار المنفرد Radio Buttons:

بعكس اداة Check Box فأن هذه الاداة لا تسمح الا بأختيار واحد فقط من بين عدة أختيارات متاحة و الصيغة القياسية لهذه الاداة هى:

```
<input name="[name]" type="radio" value="[value]">
```

حيث الخاصية name تحدد اسم الاداة او المتغير الخاص بالاداة و الخاصية value تحتوى على القيمة 1 اذا تم أختيار الاداة

مثال:

بدلا من الكود السابق قم بكتابة الكود الاتى فى الصفحة page1.htm:

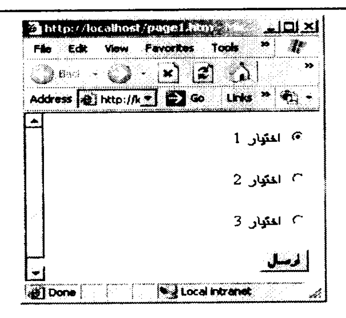
```
<html dir="rtl">
</head>

<body>
```

```
<form name="form1" method="post"
action="result.php">

     <p>
      <label>
      <input name="radiobutton" type="radio"
value="c1" checked>
      1 اختيار</label>
</p>
     <p>
      <label>
      <input name="radiobutton" type="radio"
value="c2">
      2 اختيار</label>
</p>
     <p>
      <label>
      <input name="radiobutton" type="radio"
value="c3">
      3 اختيار</label>
     </p>
     <p>
      <input type="submit" name="Submit"
value="ارسال">
         </p>
</form>
</body>
</html>
```

وتكون نتيجة IE كما بالشكل:



لاحظ ان الاختيار الاول فعالا حاول ايضا تغيير الاختييار ولاحظ انه يتم اختيار منفرد فى كل مرة .

قم الان باستبدال الكود فى الصفحة result.php بالكود الاتى:

```
<html dir = "rtl">
<?
echo $_POST["radiobutton"];
?>
</html>
```

ملاحظات:

فى هذا المثال قمنا فى الصفحة الاولى بالتأكد من ان كل ادوات الاختيار لها نفس الاسم radiobutton ولكن لكل منهم قيمة value مختلفة بحيث يمكن فى الصفحة التالية ان نقوم بكتابة سطر واحد لعرض قيمة المتغير radiobutton فلن يستطيع الزائر ان يقوم بأكثر من اختيار او بالغاء كل الاختيارات فى جميع الاحوال.

القوائم Drop down list and menu list:

وهى من الاشياء الشائعة جدا ايضا فى النماذج وتستخدم لاستقبال اختيارات الزائر و يمكن ان تسمح بعدة اختيارات او بأختيار واحد فقط كما يتضح من الصيغة القياسية:

قائمة منسدلة:

```
<select name="select">
</select>
```

قائمة متعددة الاختيارات:

```
<select name="select" size="5" multiple>
</select>
```

ويتم عن طريق الاختيار option تحديد قيمة كل اختيار كما سيتضح من المثال:

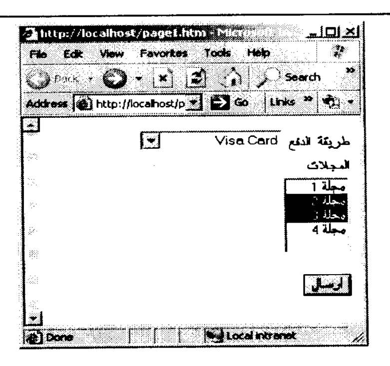
قم بكتابة الكود الاتى فى الصفحة page1.htm:

```
<html dir="rtl">
</head>

<body>
<form name="form1" method="POST"
action="result.php">
  <p>
    <label>طريقة الدفع
    <select name="select">
      <option value="visa">Visa Card</option>
      <option value="master">Master Card</option>
      <option value="american">American
Express</option>
```

```
      <option value="enroute">enRoute</option>
    </select>
</label>
<br>
   <label>المجلات</label> <br>
    <select name="select2[]" size="5" multiple>
      <option value="m1">مجلة 1</option>
      <option value="m2">مجلة 2</option>
      <option value="m3">مجلة 3</option>
      <option value="m4">مجلة 4</option>
    </select>
    </label>
</p>
  <p>
    <label>
    <input type="submit" name="Submit" value="ارسال">
    </label>
</p>
</form>
</body>
</html>
```

وعند استعراض الصفحة ببرنامج IE تكون النتيجة:



لاحظ اننا قمنا بانشاء قائمة منسدلة يستطيع الزائر ان يختار احد طرق الدفع بفتح القائمة المنسدلة ثم يمكنه عن طريق القائمة المتعددة أن يختار احد او كل المجلات التى يريد ان يشترك بها قم مثلا باختيار المجلة 2 و 3 .

قم الان بكتابة الكود الاتى فى الصفحة التالية result.php كما يلى:

```
<html dir = "rtl">
<?
echo " طريقة الدفع: ".$_POST["select"]."<br>";

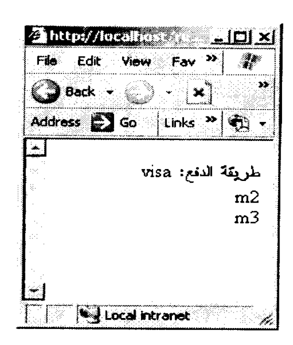
if (isset($_POST["select2"][0]))
  echo $_POST["select2"][0]."<br>";

if (isset($_POST["select2"][1]))
  echo $_POST["select2"][1]."<br>";
```

```
if (isset($_POST["select2"][2]))
  echo $_POST["select2"][2]."<br>";

if (isset($_POST["select2"][3]))
  echo $_POST["select2"][3]."<br>";
?>
</html>
```

و يجب ان تكون النتيجة كما بالشكل:



بهذه الطريقة تمكنا من معرفة اختيارات المستخدم و هناك بالطبع طرق كثيرة لاستخراج اختيارات المستخدم اهمها عن طريق الحلقات التكرارية باستخدام العبارة foreach و سنقوم بشرحها لاحقا .

لاحظ انه فى الصفحة الاولى تم تسمية القائمة بالاسم "select2[]" حتى يتم الاعلان عنها على انها مصفوفة و يمكن الوصول الى اى من عناصرها المختلفة و سيتم شرح المصفوفات بتفصيل اكثر فى الاجزاء القادمة .

لاحظ اننا استخدمنا العبارة if isset() لاختبار وجود المتغير حتى لا يقوم المعالج باظهار رسالة خطأ .

الحقل الخفى (Hidden Field):

من التقنيات الهامة جدا فهذه الحقول تسمح بتبادل المعلومات بين صفحات PHP بدون ان يراها المستخدم

هذه الحقول تأخذ الصيغة الاتية:

```
<input name="[name]" type="hidden" value="[value]">
```

يتم استبدال الخصائص name و value بالاسم و قيمة الحقل على الترتيب

مثال:

قم بوضع قيمة للحقل الخفى فى الصفحة الاولى هكذا:

```
$h_msg="قيمة سرية";
```

و لا تنس وضع علامات ?> و <? قبل كتابة كود PHP و الان قم بكتابة الحقل الخفى فى نفس الصفحة لتخزين قيمة المتغير h_msg

```
<input name="my_hidden_field" type="hidden"
value=$h_msg>
```

والان انتقل الى الصفحة التالية result.php و فيها يمكنك عرض قيمة الحقل المخفى هكذا:

```
echo $_POST["my_hidden_field"];
```

لاحظ فى الصفحة الاولى يجب ان تكون صفحة PHP و ليس HTML و يمكن اظهار عناصر HTML عن طريق العبارة "echo "html code

حقل كلمة السر password field:

ويستخدم حتى يستطيع الزائر الدخول الى بيانات خاصة بحيث لا تظهر كلمة السر لاى شخص بجانب الزائر و تظهر فيها نجوم بدلا من الحروف وهذا الحقل عبارة عن صندوق نصى بالصيغة الاتية

```
<input type="password" name="[name]">
```

نلاحظ انه تم تحديد نوعية الاداة فى الوسم input عن طريق الخاصية type و تحديد الاسم لهذه الاداة عن طريق الخاصية name

مثال:

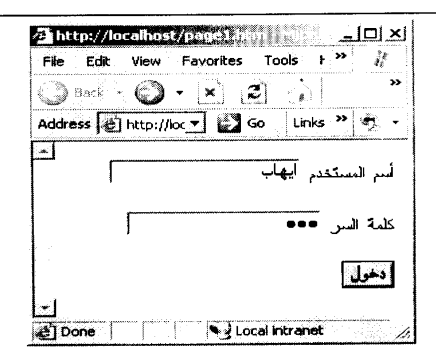
قم بكتابة الكود الاتى فى الصفحة الاولى page1.htm:

```
<html dir="rtl">
```

```
</head>

<body>
<form name="form1" method="POST"
action="result.php">
  <p>
    <label>أسم المستخدم
    <input type="text" name="user_name">
</label>
  </p>
  <p>
    <label> كلمة السر
    <input type="password" name="user_pw">
</label>
  </p>
  <p>
    <label>
    <input type="submit" name="Submit" value="دخول">
    </label>
  </p>
</form>
</body>
</html>
```

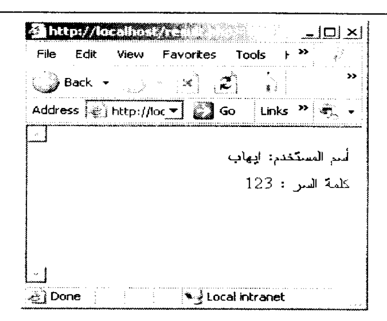
هذا الوكد يجب ان يكون نتيجته الشكل القادم:



قم الان بكتابة كود اظهار القيم فى الصفحة result.php كما يلى:

```
<html dir = "rtl">
<?
echo "أسم المستخدم: ".$_POST["user_name"]."<br>";
echo "كلمة السر : ".$_POST["user_pw"];
?>
</html>
```

بعد ان تكون قمت بادخال اسم المستخدم و كلمة السر قم بالضغط على مفتاح "دخول" و يجب ان تظهر صفحة result.php بالشكل الاتى:



ارسال البريد الالكترونى:

هناك العديد من الفوائد التى يمكن الحصول عليها من ارسال بريد تلقائى الى جهة معينة فمثلا عند وجود ملحوظة يريد الزائر ان يرسلها الى ادارة الموقع او للابلاغ عن خطأ فى جزء معين من الموقع او لارسال رابط لبريد المستخدم عن طريقه يتم تفعيل اشتراكه فى الموقع وهكذا...

ويتم استخدام الدالة ()mail لهذا الغرض كما يلى:

mail(*address,subject,message,headers*);

نلاحظ المعاملات السابقة انه يجب تحديد العنوان المرسل اليه فى المعامل address و موضوع الرسالة فى المعامل subject و الرسالة نفسها فى المعامل message و اخيرا المعامل headers التى يمكن عن طريقها تحدد بريد الراسل و ارسال ايضا نسخة طبق الاصل من الرسالة الى بريد اخر فمثلا يمكن ان يتم ذلك هكذا

"From: info@egyptbooks.net\r\nbcc:
embekheet@yahoo.com"

من المثال السابق يتم اظهار البريد info@egyptbooks.net فى الخانة from و ارسال نسخة من الرسالة الى البريد الخاص بى

مثال:

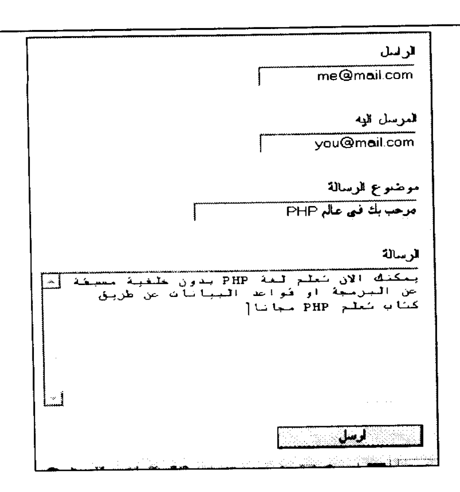
قم بكتابة الكود الاتى فى الصفحة الاولى:

```
<html dir="rtl">
</head>

<body>
<form name="form1" method="POST"
action="result.php">
  <p>
    <label>الراسل<br>
    <input name="from" type="text" size="20">
 </label>
  </p>
  <p>
    <label>المرسل اليه<br>
    <input name="to" type="text" size="20">
 </label>
  </p>
  <p>
    <label>موضوع الرسالة <br>
    <input name="sub" type="text" size="30">
```

```
</label>
  </p>
  <p>
    <label>الرسالة</label>
    <br>
    <label>
    <textarea name="msg" cols="40"
rows="10"></textarea>
</label>
  </p>
  <p>
    <label>
    <input type="submit" name="Submit" value="
ارسل            ">
</label>
  </p>
</form>
</body>
</html>
```

و تكون نتيجة الكود السابق هو نموذج ارسال الايميل كما بالشكل:



قم الان بكتابة سطر ارسال الايميل فى الصفحة result.php كما يلى:

```
<html dir = "rtl">
<?
mail($_POST["to"], $_POST["sub"], $_POST["msg"],
"From:" & $_POST["from"]);
?>
</html>
```

و عند تجربة المثال و الضغط على المفتاح "ارسل" سيتظهر رسالة الخطأ التى تدل على عدم وجود سيرفر يدعم ارسال البريد وهذا طبيعى لاننا نقوم بالتجربة على جهازنا المحلى ولكن حتى تستطيع رؤية النتيجة يجب ان تقوم بتنفيذ الكود على الانترنت.

Warning: mail() [function.mail]: SMTP server response: 550 5.7.1 Unable to relay for you@mail.com in **d:\Inetpub\wwwroot\result.php** on line **3**

one | Local intranet

الفصل الرابع

أوامر الشرط

أوامر الشرط:

الاوامر الشرطية من اساسيات اى لغة برمجة وتتحكم هذه العبارات فى خط سير البرنامج فمثلا اذا قمت بعرض عدة اختيارات للمستخدم بين ان يقوم باختيار حفظ تغييرات او الغاؤها وقام المستخدم باختيار الالغاء فيمكن التحكم فى طريقة استجابة البرنامج للمستخدم عن طريق اختبار رد فعل المستخدم .

وتستخدم العبارة IF بالصيغة الاتية لاختبار قيمة تعبير معين اذا كان صحيحا (true) ام خطأ (false) :

```
if ([condition = value]) {
//سطور تحقق الشرط
}
else{
//سطور عدم تحقق الشرط
}
```

ومن الصيغة السابقة يتبين لنا انه يتم تنفيذ السطور المحصورة بين الاقواس {} فى حالة معينة و لا يتم تنفيذ الحالتين معا فى نفس الوقت

مثال:

اذا كان قيمة المتغير e تساوى 10 فيتم طبع كلمة "النهاية"

```
<?
$e=10;
if ($e=10){
  echo "النهاية";
}
?>
```

مثال:

اذا كنت تريد السفر ولا تعرف الملابس المناسبة للجو هناك فيمكن اختبار ومعرفة اذا كان الجو دفئ او بارد باختبار درجة الحرارة كالاتى:

```
if ($temp > 24){
  echo "دفئ";
}
else {
echo "بارد";
}
```

نلاحظ من المثال السابق انه تم استخدام العلامات الحسابية المنطقية (<) لاختبار درجة الحرارة و تعنى اكبر من و يوجد ايضا اكثر من معامل يمكن استخدامهم حسب الوظائف الموضحة :

المعامل	الوظيفة
=	لمقارنة قيمتين بالتساوى
val < val2	القيمة val1 اصغر من القيمة val2
val > val2	القيمة val1 اكبر من القيمة val2
val <= val2	القيمة val1 اصغر من او تساوى القيمة val2
val >= val2	القيمة val1 اكبر من او تساوى القيمة val2
val <> val2	القيمة val1 لاتساوى القيمة val2

مثال:

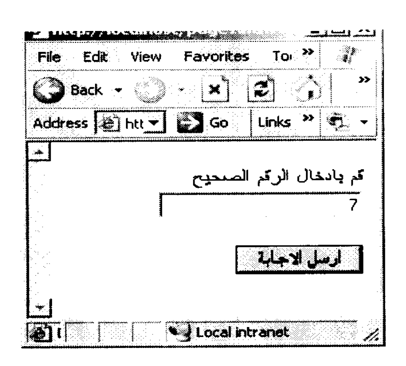
سنقوم الان بتنفيذ لعبة بسيطة وهى تخمين رقم عشوائى من 1 الى 10 فأذا كان الرقم الذى خمنه الزائر اقل من الرقم الصحيح يجب ان تظهر رسالة لتدل على ذلك و بالمثل اذا كان الرقم أكبر أما أذا كان الرقم صحيح فيجب أظهار رسالة ان الزائر قد كسب اللعبة .

قم الان بكتابة الكود الاتى:

```
<html dir="rtl">
</head>

<body>
<form name="form1" method="POST"
action="result.php">
  <p>قم بادخال الرقم الصحيح<br>
    <label>
    <input type="text" name=" user_no">
</label>
  </p>
  <p>
    <label>
    <input type="submit" name="Submit" value=" ارسل
الاجابة">
</label>
  </p>
</form>
</body>
</html>
```

و يكون الشكل كما يلى:



قم الان بكتابة الكود الاتى فى الصفحة result.php و التى ستحتوى على الكود الفعلى لحساب الرقم العشوائى:

```
<html dir = "rtl">
<?
$correct_no = rand(1,10);

if ($_POST["user_no"] > $correct_no){
echo "الرقم الذى قمت بتخمينه خطأ وهو أكبر من الرقم الصحيح
".$correct_no."<br>";

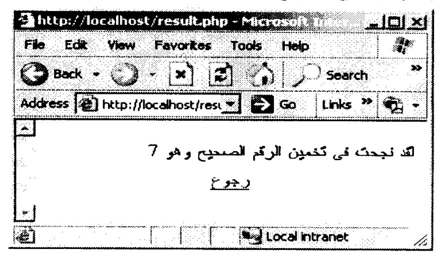
}
elseif ($_POST["user_no"] < $correct_no){
```

```
echo "الرقم الذى قمت بتخمينه خطأ وهو أصغر من الرقم الصحيح
".$correct_no."<br>";
}
else {
  echo "لقد نجحت فى تخمين الرقم الصحيح وهو
".$correct_no."<br>";
}

//echo "أسم المستخدم: ".$_POST["user_name"]."<br>";
//echo "كلمة السر : ".$_POST["user_pw"];

?>
 <div align="center"><a
href="javascript:history.back()">رجوع</a></div>
</html>
```

و عند تجربة اللعبة يجب ان تظهر النتيجة بالشكل:



ولقد نجحت اخيرا فى تخمين الرقم الصحيح بعد عناء ولكن لا تقلق أذا لم تصل الرقم الصحيح بعد عدة محاولات حاول مرة اخرى !

لاحظ من الكود المكتوب فى صفحة result.php استخدام الدالة rand التى تقوم بارجاع الرقم الصحيح العشوائى المحصور بين 1 و 10 و يتم مقارنتها عن طريق العبارة if و elseif لتدل على عدم تحقق الشرط السابق ثم اخيرا العبارة else التى تدل على عدم تحقق اى من الشروط السابقة .

لقد قمت ايضا بوضع رابط يستطيع الزاعر عن طرقه الرجوع مرة اخرى الى الصفحة السابقة و تم هذا باستخدام لغة الجافا سكريبت عن طريق العبارة history.back() وهى من الاوامر المفيدة عمليا .

معاملات خاصة:

اوجزنا فيما سبق المعاملات الحسابية المعروفة مثل التساوى و اكبر من و اصغر من و لكن توجد ايضا علامات خاصة تعطى وظائف وفوائد اكبر عند استخدامها وهى:

علامات المقارنة (===,==):

ذكرنا سابقا ان علامة التساوى المنفردة (=) تستخدم لالحاق قيمة بمتغير مثل العبارة:

```
$var=14;
```

و لكن هذه العلامات تعبر عن المقارنة بين قيمتين و تستخدم مع العبارة IF فمثلا لمقارنة ما اذا كان المتغير s1 يساوى المتغير s2 يجب كتابته بالصورة الاتية:

```
$s1=22;
$s2="22";
```

```
if ($s1 == $s2) {
  echo "القيم متساوية";
} else {
  echo "القيم غير متساوية";
}
```

لاحظ اننا قمنا بالمقارنة بين نوعين مختلفين من المتغيرات حرفى و عددى ورغم ذلك فان تقدير نتيجة المقارنة يؤدى الى حالة التساوى لان استخدام العلامتين == لا يفرق بين انواع المتغيرات و لكن القيم فقط

أما اذا كتبنا عبارة IF كما يلى :

```
if ($s1 === $s2) {
```

فان تقدير النتيجة ستكون عدم التساوى لان العلامات === تفرق بين القيم و الانواع معا

المعاملين (<>) و (=!) :

العلامتين يعبران عن عدم التساوى و يتم استخدامهم كما بالمثال الاتى:

```
if ($s1 <> $s2) { }
if ($s1 != $s2) { }
```

و السطرين السابقين يدلان على نفس المعنى وهو "أذا كان المتغير s1 لا يساوى المتغير s2 فنفذ السطور القادمة"

المعاملات المنطقية:

هذه المعاملات هى ثلاثة (AND, OR, NOT) وهى من علامات المقارنة فى جميع لغات البرمجة و يتم استخدامهم كما يلى
المعامل AND للتعبير عن ضرورة تحقق شرطين قبل و بعد المعامل حتى يتم تقدير العبارة كلها على انها (true) فمثلا يجب ان نكتب الشرط كما يلى:

```
if ($s1 = 20 AND $s2 = 20) { }
```

اذا تحقق الشرطين معا يمكن تنفيذ السطور القادمة أما اذا كان احد الشرطين لا يتحقق (تكون نتيجته false) فأن العبارة كلها لا تحقق (تكون كلها false) ويتم تنفيذ الجزء else.

بالمثل هناك مثيل لعلامة الاضافة AND و هى العلامتين "&&" وهى تؤدى نفس المعنى و نفس النتيجة

```
if ($s1 = 20 && $s2 = 20) { }
```

و نلاحظ هنا ان لغة PHP جمعت بين عناصر اللغات عالية المستوى مثل الفيجوال بيزيك و عناصر لغة السى بلس بلس

المعامل أو "||" , "OR":

عكس المعامل AND فهذا المعامل يكفى تحقق احد شروطه حتى يتم تقدير العبارة كلها على انها متحققة (true)

مثال:

```
$t=45;
$w=80;

if ($t = 40 || $t = 70 || $w=80 || $w=23) {
  echo "النتيجة متحققة";
}
```

بالمثل يمكن كتابة العبارة السابقة كما يلى:

```
$t=45;
$w=80;

if ($t = 40 OR $t = 70 OR $w=80 OR $w=23) {
  echo "النتيجة متحققة";
}
```

و يمكن الحصول على العلامتين "||" بالضغط على المفتاح Shift + المفتاح "\" مرتين

المعامل "!":

يوجد باللغات الاخرى المعامل NOT الذى يساوى المعامل "!" من حيث الوظيفة ولكن لغة PHP على تحتوى على المعامل NOT و يجب كلما أمكن استخدام المعاملات المنطقية الرمزية ("&&" و "=!" و "||") كلما امكن لانها اسرع فى الترجمة لان هذه اللغة مكتوبة عن طريق لغة ++C .

عندما يجد المعالج هذا المعامل يفهم ان الشرط كله يتحقق اذا كان الشرط بعد علامة **"!"** لا يتحقق

مثال:

```
$pass="987654";
if !($pass == "987654") {
 echo "كلمة السر خطأ";
}else {
 echo "كلمة السر صحيحة";
 }
```

من المثال السابق نرى ان نتيجة الشرط هى التحقق و طباعة كلمة "كلمة السر صحيحة" لان العلامة **"!"** قد نفت تحقق الشرط الذى بعدها .

و يمكن استخدامها ايضا لاختبار قيمة متغير شرطى من نوع Boolean كما يلى:

```
if !($b) { }
```

وهذا معناه اذا كان المتغير $b غير متحقق (not true) يتم تنفذيذ السطور التالية

يمكن المزج بين الشروط عن طريق الاقواس بحيث يتم التحقق من حدوث شرط مركب فى نفس السطر

```
$x= 8;
$y=7;
$z=5;
if ($x == 10) || ($x == 8) && ($y != 6) && ($z <= 5) {
  echo "تحقق الشرط";
}
```

نلاحظ من العبارة السابقة كيف تم تحققيق الشرط بالكامل لان اجزاؤه المختلفه تحققت وتم ذلك كما يلى:

1- ($x == 10) || ($x == 8)

تكون نتيجته (true) لان المتغير x يساوى 8

2- ($y != 6) && ($z <= 5)

نرى ايضا تحقق هذا الجزء لان المتغير y لا يساوى 6 بل 5 و المتغير z يساوى بالفعل القيمة 5 فأذا كان الجزئين يتم تحقيقهم فأن العبارة بالكامل يتم تقديرها على انها (true) كما يتضح من الرسم التزضيحى:

(true) && (true) ➔ (true)

عبارة IF المتعددة:

يمكن ان تتعدد العبارة IF بحيث يتم التحقق من عدة شروط مختلفة وعلى اساس تقييمها يتم تنفيذ سطور معينة من الكود تناسب كل حالة و يتم ذلك بالصيغة الاتية:

```
if (شرط 1) {
//سطور كود
}elseif (شرط 2) {
//سطور كود
}else {
//سطور كود
}
```

مثال:

اذا اردت اختبار درجة الطالب فى احد الامتحانات فأذا حصل على 90 فأكثر يتم طبع "امتياز" و أذا حصل على درجة ما بين 60 و 90 يتم طباعة "جيد" و أذا حصل على أقل من 60 يتم طباعة "رسوب" و يمكن تنفيذ ذلك كما يلى:

```
$deg=80;
if ($deg > 90) {
  echo "امتياز";
}elseif ($deg >= 60 && $deg <= 90) {
  echo "جيد";
}else {
  echo "رسوب";
}
```

يجب حتى يتحقق الجزء else الا يتحقق اى من الشروط السابقة :

كما يمكن تنفيذ تداخل بين عبارات IF بحيث لا يتم قطع اى حلقة IF لحلقة اخرى مثيلتها فمثلا المثال التالى صحيح:

```
if ("شرط") {

  if ("شرط") {
   "  أجراء معين"
  }else {
   "اجراء اخر"
  }
}
```

هام جدا:

يجب ان تكون الاقواس المفتوحة } بعدد الاقواس المقفلة { كما يتضح من المثال السابق و يجب تنظيم الاقواس اسفل بعضها البعض مع ازاحة العبارات الفرعية الى اليمين مسافتين على الاقل حتى يمكنك قراءة الكود بسهولة و تصحيح اى خطأ قد يوجد نتيجة للسهو فأذا كتبت الكود السابق كما يلى سيكون صعب فى الفهم او اكتشاف الاخطاء.

```
if ("شرط") {if ("شرط") {"أجراء معين"} else{ "اجراء اخر"}}
```

العبارة Switch:

تقوم العبارة switch بوظيفة مماثلة للعبارة IF و لكن بعكس العبارة IF التى يمكن عن طريقها تقدير نتيجة عدة متغيرات تقوم العبارة switch بتقدير نتيجة متغير واحد و يتم بناء على القيم المختلفة للمتغير تنفيذ اجراءات مختلفة من الكود

ولا يمكن اختبار صحة شرط باستخدام العبارة switch و لكن يتم استخدامها للتعامل مع عدة قيم متوقعة من متغير معين .

و تأخذ العبارة الصيغة القياسية الاتية:

```
switch ( expression )
{
case result1:
// execute this if expression results in result1
break;
case result2:
// execute this if expression results in result2
break;
default:
// execute this if no break statement
// has been encountered hitherto
}
```

شرح العبارة:

يتم اختبار قيمة متغير بسيط (حرفى او عددى) بدلا من القيمة expression ثم يتم تنفيذ اول حالة من حالات case فأذا تحقق تم تنفيذ الكود الذى يوجد بعدها ثم يصل المعالج الى السطر الذى يوجد به العبارة ;break فيتم الخروج من الجزء switch و يتم تنفيذ السطور التى توجد بعده .

أما أذا لم تتحقق الحالة الاولى يتم تنفيذ الحالة الثانية و هكذا فأذا لم تتحقق اى من الحالات السابقة فيتم تنفيذ الكود الذى يوجد بعد الحالة default .

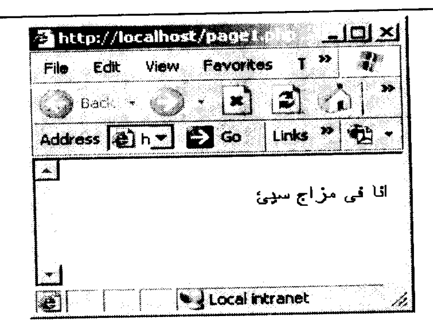
هام جدا:

اذا لم تكتب العبارة ;break بعد كل نهاية حالة فأن المعالج سيستمر فى تنفيذ بقية الحالات حتى اذا تحقق احداها وفى معظم الاحيان انت لاتريد تحقيق ذلك.

مثال:

```
<html>
<body>
<?php
$mood = "حزين";
switch ( $mood ){
  case "سعيد":
    print "انا فى مزاج جيد";
    break;
  case "حزين":
    print "انا فى مزاج سيئ";
    break;
  default:
    print " انا لست حزينا او سعيدا بل: $mood";
  }
?>
</body>
</html>
```

و تكون نتيجة تنفيذ الكود السابق هى :



لاحظ هنا اننا استخدمنا العبارة print وهى تعطى نفس وظيفة echo تقريبا

الشرط عن طريق الاستفهام "?":

و يتم ذلك بالصيغة الاتية:

```
( expression
)?returned_if_expression_is_true:returned_if_expression_is
_false;
```

مثال:

```
$mood = "sad";
$text = ( $mood=="سعيد" )?"انا فى مزاج جيد":"لست فى مزاج
 جيد بل $mood";
print "$text";
```

لا حظ مما سبق انه يتم تنفيذ الاجراء بعد علامة الاستفهام مباشرة اذا كان الشرط صحيحا و الا يتم تنفيذ الاجراء الذى يلى علامة التنقيط ":" كما بالمثال.

ملحوظة:

اذا كنت تعطى المستخدم امكانية ادخال قيمة ضرورية بحيث تخاف ان يقوم المستخدم بادخال وسوم HTML فى عملية الادخال مما يؤدى الى ان يتعامل معها المعالج على انها كود HTML فيمكن استخدام الدالة HtmlSpecialChars التى تقوم بحذف العلامات الخاصة بكود HTML بحيث يتم طباعتها كما هى .

مثال:

```
$last_name = htmlspecialchars($last_name);
```

بهذه الطريقة اذا كان المتغير last_name يحتوى على علامات مثل "<" او ">" فيتم طباعتها كما هى .

الفصل الخامس

الحلقات التكرارية

الحلقات التكرارية LOOPS:

تعطينا لغة PHP امكانية تنفيذ مجموعة من التعليمات لعدد من المرات ويتم ذلك عن طريق عبارات الحلقات التكرارية ويكون للحلقة شرط معين اذا لم يتحقق تستمر الحلقة فى التكرار حتى يتم الوصول الى تحقيق الشرط فيتم انهاء الحلقة او قد تستخدم عبارات اخرى تؤدى الى الخروج الاجبارى من الحلقة كما سنرى:

العبارة While:

و الصيغة القياسية لهذه العبارة هى:

```
while (شرط الايقاف){
//يتم تكرار هذا الجزء
}
```

معنى العبارة السابقة انه يتم تكرار الحلقة طالما الشرط متحقق (true) و يجب ان تقوم فى داخل الحلقة بتغيير هذا الشرط الى false والا يتم تنفيذ الحلقة الى مالانهاية .

مثال (تجاهل ارقام السطور هى للتوضيح فقط):

```
1: <html dir="rtl">
2: <head>
3: <titel>مثال للحلاقات التكرارية</title>
4: </head>
5: <body>
```

```
6: <?php
7: $counter = 1;
8: while ( $counter <= 12 )
9: {
10: print "$counter × 2 = ".($counter*2)."<br>";
11: $counter++;
12: }
13: ?>
14: </body>
15: </html>
```

نلاحظ مما سبق اننا قمنا بتعريف متغير فى السطر رقم 7 واعطيناه القيمة المبدئية 1 ثم بدئنا الحلقة و وضعنا الشرط لهذه الحلقة انها لاتكرر الا اذا كان العداد اقل من او يساوى 12 اى ان الحلقة ستتكرر 12 مرة .

ثم وضعنا عبارة print بسيطة فى السطر رقم لتقوم بطبع ناتج رقم العداد مضروبا فى 2 .

واخيرا تم زيادة قيمة العداد بقدار واحد صحيح فى كل مرة يتم فيها تكرار الحلقة وتم ذلك عن طريق العبارة الشهيرة لزيادة العداد وهى :

```
$var++;
```

وهذه العلامة ++ تساوى العبارة :

```
$var = $var + 1;
```

الحلقة Do..While:

ولها الصيغة:

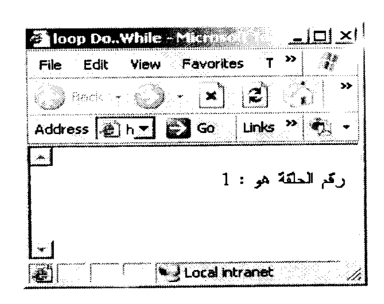
```
do {
// سطور الكود التى تتكرر
}
while ( شرط ايقاف الحلقة );
```

هذه الحلقة تعتبر مثل حلقة while ولكنها مقلوبه رأسا على عقب و الفرق الاساسى بينهم انه يتم فى هذه الحلقة تنفيذ اول تكرار قبل ان يتم اختبار الشرط اذا كان متحقق ام لا

مثال:

```
<html dir="rtl">
<head>
<title>loop Do..While</title>
</head>
<body>
<?php
$num = 1;
do
{
print " رقم الحلقة هو: $num<br>\n";
$num++;
}
while ( $num > 200 && $num < 400 );
?>
</body>
</html>
```

تنفيذ الكود السابق هو النتيجة:



و الان هل تعرف لماذا تم تنفيذ الحلقة مرة واحدة فقط ؟

من المثال السابق يتضح ان الحلقة do..while قمانت بالتكرار قبل ان تقوم بتقدير قيمة متغير الحلقة num و لايتحقق التقدير لذلك فأن الحلقة تتنهى عند هذا الحد ؟ ، حاول الان تغيير شرط الحلقة الى

```
while (num <400);
```

وراقب ماذا يحدث

الحلقة For:

وهى تشبه الحلقة While تماما فيما عدا انها امنه اكثر و يتضح ذلك من الصيغة انقياسية لهذه الحلقة:

for (***variable assignment***; ***test expression***; ***variable increment***)
{

```
// الكود الذى يتم تكراره
}
```

تذكر من المثال السابق الخاص بالحلقة While كيف قمنا اولا بتعريف متغير و اعطاؤه قيمة مبدئية ثم كتبنا الشرط الخاص بالمتغير بعد العبارة while ثم قمنا داخل الحلقة بزيادة قيمة المتغير حتى تنتهى الحلقة عند حد معين فكل ذلك يمكن تنفيذه مع الحلقة FOR فى سطر واحد مما يجعل احتمالات ان تنسى زيادة المتغير مثلا منعدمة مما قد يؤدى الى حلقة لانهائية .

مثال:

```
<html dir="rtl">
<head>
<title>FOR loop</title>
</head>
<body>
<?php
for ( $counter=1; $counter<=12; $counter++ )
{
print "$counter × 2 = ".($counter*2)."<br>";
}
?>
</body>
</html>
```

لاحظ ان هذا المثال يعطى نفس نتيجة المثال الخاص بالحلقة While فيتم تمهيد قيمة مبدئية للمتغير العداد فى اول معامل من معاملات For ثم يتم تحديد الشرط الذى تتوقف عنده الحلقة ثم يتم زيادة العداد فى المعامل الاخير

الخروج من الحلقات:

يمكن الخروج من الحلقات بطريقة اجبارية عند تحقق شرط معين داخلى باستخدام عبارة IF مدلا بحيث لايتم استكمال عملية التكرار ويتم ذلك هكذا:

```
1: <html dir="rtl">
2: <head>
3: <title>الخروج من الحلقات</title>
4: </head>
5: <body>
6: <?php
7: $counter = – 4;
8: for ( ; $counter <= 10; $counter++ )
9: {
10: if ( $counter == 0 )
11: break;
12: $temp = 4000/$counter;
13: print "4000 divided by $counter is...
$temp<br>";
14: }
15: ?>
16: </body>
17: </html>
```

افترض مثلا انك تستقبل قيمة من الزائر وتقوم باجراء عمليات التكرار بناء على هذه القيمة وماذا اذا كتب الزائر قيمة اقل من الصفر فى هذه الحالة يجب ان نختبر قيمة العداد بحيث لا يتم القسمة على الصفر و يتم ذلك عن طريق عبارة IF فأذا تحققت يتم تنفيذ العبارة ;break ويتم الخروج من الحلقة فى السطر 11

لاحظ انه يمكن عدم كتابة اى معامل من معاملات For ولكن يجب ترك الفاصلة المنقوطة مكانها ";"

هناك طريقة اخرى لتفادى تنفيذ بعض الحلقات التكرارية و هى العبارة Continue; وفيها بدلا من ان يتم ايقاف الحلقة نهائيا و الخروج منها يتم تفادى تكرار بعض الحلقات بناء على شرط معين كما سنرى:

مثال:

```
<html dir="rtl">
<head>
<title> العبارةContinue</title>
</head>
<body>
<?php
$counter = – 4;
for ( ; $counter <= 10; $counter++ )
{
if ( $counter == 0 )
continue;
$temp = 4000/$counter;
print "4000 divided by $counter is... $temp<br>";
}
?>
</body>
</html>
```

هنا فى حالة اذا كان العداد صفر يتم تفادى تنفيذ الحلقة حتى تكون قيمة العداد غير صفرية و يذهب المعالج الى العبارة For و العبارة while ليعيد تقييم العداد مرة اخرى ولا يتم تنفيذ السطور التى تأتى بعد ;Continue .

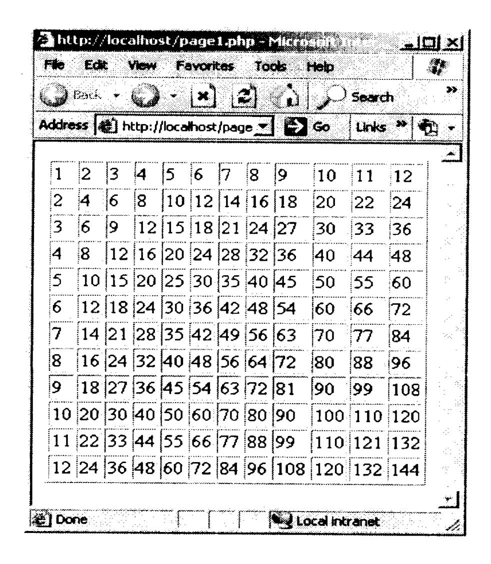
التداخل بين الحلقات:

يمكن اجراء تداخل لحلقة داخل اخرى بشرط الا يتقاطعا و هذا مفيد اذا كنت تريد انشاء جدول HTML داخل الصفحة و يمكن تنفيذ ذلك كما يلى:

```
<html>
<head>
<title>تداخل الحلقات</title>
</head>
<body>
<?php
print "<table border='1'>\n";
for ( $y=1; $y<=12; $y++ )
{
print "<tr>\n";
for ( $x=1; $x<=12; $x++ )
{
print "\t<td>";
print ($x*$y);
print "</td>\n";
}
print "</tr>\n";
}
print "</table>";
?>
</body>
```

```
86
22: </html>
```

بعد تنفيذ هذا الكود يتم تكوين جدول فى الصفحة يتكون من 12 صف و 12 عمود كما بالشكل:



1	2	3	4	5	6	7	8	9	10	11	12
2	4	6	8	10	12	14	16	18	20	22	24
3	6	9	12	15	18	21	24	27	30	33	36
4	8	12	16	20	24	28	32	36	40	44	48
5	10	15	20	25	30	35	40	45	50	55	60
6	12	18	24	30	36	42	48	54	60	66	72
7	14	21	28	35	42	49	56	63	70	77	84
8	16	24	32	40	48	56	64	72	80	88	96
9	18	27	36	45	54	63	72	81	90	99	108
10	20	30	40	50	60	70	80	90	100	110	120
11	22	33	44	55	66	77	88	99	110	121	132
12	24	36	48	60	72	84	96	108	120	132	144

و يتضح من المثال ان المتغير x يتم تكراره 12 مرة فى كل تكرار واحد من المتغير y وبذلك يتم طبع الخلايا فى كل صف من صفوف الجدول عن طريق الوسم TD .

الفصل السادس

الدوال

الدوال Functions:

استخدام الدوال هام جدا حتى لا يكون البرنامج كبير الحجم وصعب الفهم كما تظهر اهميتها عند تكرار الحاجة لاداء نفس الوظيفة مرة اخرى .

ويمكنك ان تفكر فى الدوال مثل الالة التى يدخل اليها المادة الخام و تقوم بعد المعالجة بإخراج الناتج فمثلا اذا اردت ان تقوم بخبز كعكة مرة واحدة فأنك سوف تفعلها بنفسك ولكن اذا اردت ان تخبز الكعكة الاف المرات فأنك يجب ان تقوم بشراء مكينة لتحضير الكعك و بالمثل فأن تقرير استخدام الدوال او لا هو الحاجة لاستخدام نفس الكود العديد من المرات .

لذلك عندما تقوم بنداء دالة فأنك تعطيها قيمة او عدة قيم و تقوم هى بعد معالجة خاصة لهذه القيم بارجاع قيمة معينة يمكنك ان تقوم باستخدامها فى الكود .

والدوال تنقسم الى نوعين دوالة مبنية فى اللغة ذاتها و دوال تقوم انت نفسك بتعريفها .

مثال لدالة جاهزة (built-in):

سنقوم الان باستخدام الدلة abs() التى تقوم بأخذ قيمة عشرية (رقم سالب مثلا) وتقوم الدالة بارجاع القيمة المطلقة (الصحيحة) لهذا العدد ويتم تنفيذ ذلك كما يلى:

```
1: <html dir="rtl">
2: <head>
3: <title>دالة الاعداد الصحيحة</title>
4: </head>
5: <body>
6: <?php
7: $num = – 321;
8: $newnum = abs( $num );
```

```
9: print $newnum;
10: // prints "321"
11: ?>
12: </body>
13: </html>
```

كما يتضح فان المتغير newnum سيحتوى فى السطر 8 على القيمة الصحيحة للعدد (321-)

تعريف الدوال:

الصيغة القياسية لتعريف الدوال هى:

```
function some_function( $argument1, $argument2 )
{
// function code here
}
```

حيث argument1 و argument2 هى المعاملات او المادة الخام التى نقوم بتمريرها الى الدالة ومن ثم تقوم الدالة بعد المعالجة بارجاع القيمة النهائية

مثال:

فى هذا المثال سنقوم بتعريف دالة و النداء عليها كما يلى:

```
<html dir="rtl">
<head>
<title>تعريف دالة</title>
</head>
```

```
<body>
<?php
function bigtitle()
{
print "<h1>مرحب بك</h1>";
}
bigtitle();
?>
</body>
</html>
```

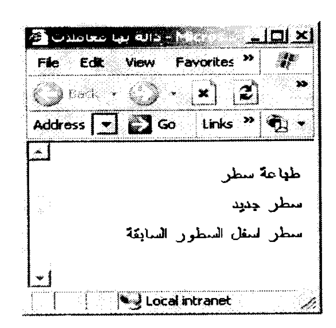
كما سبق فى المثال فقد قمنا بتعريف الدالة وفى هذه الحالة لاتأخذ اى معاملات ولكن عند النداء عليها تقوم بطبع كلمة "مرحب بك" بخط كبير .

سنقوم الان بكتابة مثال لدالة بها معاملات:

```
<html dir="rtl">
<head>
<title>دالة بها معاملات</title>
</head>
<body>
<?php
function print_line_msg( $txt )
{
print ("$txt<br>\n");
}
print_line_msg ("طباعة سطر");
print_line_msg ("سطر جديد");
print_line_msg ("سطر اسفل السطور السابقة");
?>
```

```
</body>
</html>
```

و يجب ان تكون نتيجة تنفذيذ السطور السابقة هى الشكل:



نلاحظ من هذا المثال اننا قمنا باعطاء الدالة قيم مختلفة وفى كل مرة تقوم الدالة بطباعة النص و ادراج سطر جديد اسفل النص عن طريق كود HTML "
 /n"

سنقوم الان بتعريف دالة تقوم بارجاع قيمة معينة:

```
<html>
<head>
<title>دالة تعطى قيمة</title>
</head>
<body>
<?php
function addNums( $firstnum, $secondnum;
```

```
8: {
9: $result = $firstnum + $secondnum )
10: return $result;
11: }
12: print addNums(3,5);
13: // النتيجة ستكون 8
14: ?>
15: </body>
16: </html>
```

من المثال البسيط السابق نرى ان الدالة addNum تقوم باضافة رقم اول معامل الى رقم ثانى معامل وتقوم بطبع الناتج و يتم تحقيق ذلك فى السطر رقم 10 الذى يحتوى على العبارة return والتى تقوم بتخزين القيمة للمتغير result فى الدالة .

مدى المتغيرات:

المتغيرات المعلن عنها داخل الدوال تبقى معرفة فقط داخل حدود الدالة ولا يمكن الوصول اليها خارجها وهذا مفيد لانه يحميك من ان تقوم بتغيير قيمة متغير على سبيل الخطأ خارج الدالة .

والمثال الاتى يقوم بتوضيح عدم امكانية الوصول الى المتغير المحلى المعرف داخل الدالة:

```
1: <html dir="rtl">
2: <head>
3: <title>مدى المتغير المحلى</title>
```

```
</head>
<body>
<?php
function test()
{
$testvariable = "this is a test variable";
}
print "test variable: $testvariable<br>";
?>
</body>
</html>
```

نرى فى هذا المثال ان برنامج IE لا يقوم بطباعة شيئ؛ لان المتغير نفسه غير موجود خارج الدالة .

تعريف المتغيرات العالمية Global Variables:

بعكس المتغيرات المحلية يمكن الوصول الى المتغيرات العالمية من اى مكان فى البرنامج .

مثال:

```
<html dir="rtl">
<head>
<title>مدى المتغيرات</title>
</head>
<body>
<?php
$life = 42;
function meaningOfLife()
{
```

```
print "The meaning of life is $life<br>";
}
meaningOfLife();
?>
</body>
</html>
```

تكون نتيجة المثال السابق طباعة لاشئ لان المتغير life ليس معرفا داخل الدالة فلا تكون له القيمة 42 ولكن لكى تستطيع ان تصل الى قيمة من خارج الدالة يجب تمريرها عن طريق المعاملات .

ويمكن ايضا الوصول الى متغيرات الخارجية ولكن يجب فى هذه الحالة استخدام العبارة global كما يلى:

```
<html>
<head>
<title>Listing 6.8</title>
</head>
<body>
<?php
$life=42;
function meaningOfLife()
{
global $life;
print "The meaning of life is $life<br>";
}
meaningOfLife();
?>
</body>
</html>
```

نرى من المثال السابق ان المتصفح يقوم بطباعة قيمة المتغير life لانه تم الاعلان عنه بالعبارة global و تكون النتيجة هى طباعة الرسالة:

The meaning of life is 42

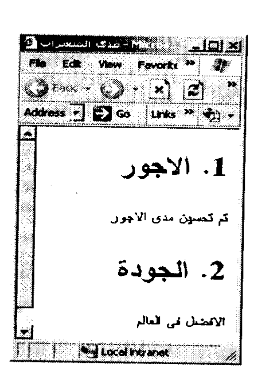
ذكرنا ان المتغير المحلى المعرف داخل الدالة يتم استخدامه ثم تنتهى حياته عند انتهاء تنفيذ الدالة ولاكن يمكن ايضا الحفاظ على قيمة المتغير داخل و خارج الدالة بحيث اذا تم تنفيذ الدالة مرة اخرى يتم تذكر اخر قيمة للمتغير و يتم ذلك بطريقتين أما باستخدام العبارة global او باستخدام العبارة static .

فيما يلى كيفية اداء ذلك عن طريق العبارة global:

```
<html dir="rtl">
<head>
<title>مدى المتغيرات</title>
</head>
<body>
<?php
$num_of_calls = 0;
function second_item( $txt )
{
global $num_of_calls;
$num_of_calls++;
print "<h1>$num_of_calls. $txt</h1>";
}
second_item("الاجور");
print("تم تحسين مدى الاجور<p>");
```

```
second_item ("الجودة");
print("الافضل فى العالم<p>");
?>
</body>
</html>
```

وتكون نتيجة المثال الشكل الاتى:



و يمكن تنفيذ ماسبق عن طريق العبارة static كما يلى:

```
<html dir="rtl">
<head>
<title>Static</title>
</head>
<body>
<?php
function second_item ( $txt )
```

```
{
static $num_of_calls = 0;
$num_of_calls++;
100
print "<h1>$num_of_calls. $txt</h1>";
}
second_item ("الاجور ");
print("تم تحسين مدى الاجور <p>");
second_item ("الجودة ");
print("الافضل فى العالم <p>");
?>
</body>
</html>
```

وستكون النتيجة مثل المثال السابق تماما ولكن باسلوب افضل

الحاق المعامل بقيمة افتراضية:

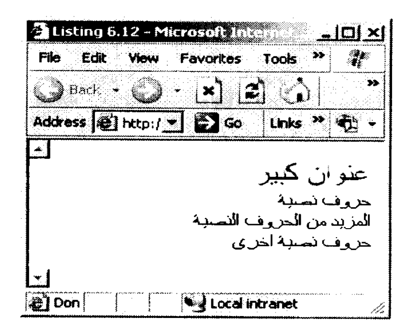
يمكنك ان تقوم بوضع قيمة افتراضية للمعامل الذى يتم تمريره الى الدالة بحيث لا تحتاج الى وضعه ضمن معاملات الدالة عند النداء عليها .

مثال:

```
<html dir="rtl">
<head>
<title>معاملات الدوال</title>
</head>
<body>
```

```
<?php
function fontWrap( $txt, $size=3 )
{
print "<font
size=\"$size\"face=\"Helvetica,Arial,Sans-
Serif\">$txt</font>";
}
fontWrap("عنوان كبير<br>",5);
fontWrap("حروف نصية<br>");
fontWrap("المزيد من الحروف النصية<br>");
fontWrap("حروف نصية اخرى<br>");
?>
</body>
</html>
```

يمكنك ان ترى من المثال السابق اننا لم نحتاج الى الحاق قيمة للمعامل size ويتم استخدام القيمة الافتراضية 3 فى هذه الحالة وتكون النتيجة كما بالشكل:



تمرير المتغيرات المرجعية:

افتراضيا فى الحالات العادية يتم تمرير المتغيرات فى معاملات الدوال عن طريق قيمة المتغير اى ان اى تغيير فيى قيمة المتغير داخل الدالة لا يؤثر على قيمة المتغير خارجها وهذا يسمى التمرير بالقيمة او (variable by value) اما اذا اردنا ان نتحكم فى قيمة المتغير و نقوم بتغييرها حتى بعد انتهاء عمل الدالة يجب ان نستخدم هذه التقنية التى تسمى (variable by reference) ويتم ذلك كما بالمثال:.

```
1: <html dir="rtl">
2: <head>
3: <title>variables by reference</title>
4: </head>
5: <body>
6: <?php
7: function addFive( &$num )
8: {
9: $num += 5;
10: }
11: $orignum = 10;
12: addFive( $orignum );
13: print ( $orignum );
14: ?>
15: </body>
16: </html>
```

هل تستطيع الان ان تخمن ماهى نتيجة المثال السابق ؟ سيتم ببساطة تغيير قيمة المتغير orignum فى السطر 12 وسيتم طبع النتيجة 15 فى المتصفح.

الفصل السابع

المصفوفات

المصفوفات Arrays:

عرفنا سابقا ان المتغيرات تستطيع ان تحمل قيمة منفردة من البيانات سواء العددية او الحرفية و اذا احتجت الى اكثر من متغير فانك تقوم بتعريفه ولكن على عكس المتغيرات فانك اذا قمت بتعريف المصفوفة فتستطيع ان تقوم بتخزين قيم كثيرة فى نفس الوقت و تستطيع الوصول الى كل قيمة عن طريق فهرس للمصفوفة .

كما تستطيع ترتيب هذه البيانات المخزنة فى المصفوفة ترتيبا ابجديا او عدديا او ترتيب خاص يلائمك ايضا يمكنك عن طريق الحلقات المرور على كل القيم داخل المصفوفة وقرائتها وتعديلها اذا لزم الامر .

يبداء فهرس المصفوفة دائما برقم صفر لذلك فدائما اخر رقم فى المصفوفة يقل بواحد عن عدد العناصر فى المصفوفة .

تعريف المصفوفات:

- استخدام الدالة ()array:

يمكن استخدام هذه الدالة لتعريف مصفوفة بالطريقة الاتية:

```
$myBooks = array("الهاكرز","تعلم دلفى 8","تعلم اكسيس");
```

يمكنك الان اذا اردت طباعة ***الثانى*** فى المصفوفة ان تستخدم العبارة:

```
print $myBooks[1];
```

- **استخدام الاقواس المربعة:**

يمكنك ايضا تعريف مصفوفة بطريقة سهلة عن طريق الاقواس كما يلى:

```
$myBooks[] = "تعلم اكسيس";
$myBooks[] = "تعلم دلفى 8";
$myBooks[] = "الهاكرز";
```

وبنفس الطريقة يمكنك الوصول الى العنصر الثالث مثلا عن طريق العبارة:

```
print $myBooks[2];
```

ويمكن ايضا استخدام هذه الطريقة مع الطريقة السابقة لاضافة عناصر جديدة الى المصفوفة بعد تعريفها

المصفوفات المترابطة:

يمكن ايضا تعريف فهرس المصفوفات عن طريق قيم حرفية وليست عددية فقط و يتم ذلك بطريقتين:

- **استخدام الدالة array():**

عند تعريف المصفوفة المترابطة يجب تعريف المفتاح و القيمة للمصفوفة كما يلى:

```
$employee = array (
name=>"ايهاب",
occupation=>"محاسب",
age=>20,
"Department no"=>"5C"
);
```

يتضح من المثال السابق كيفية انشاء مصفوفة لتخزين بيانات الموظفين و ليس من الضرورى ادراج علامتى تنصيص لعناصر هذه المصفوفة الا اذا كانت تتكون من اكثر من كلمة وفى بعض اصدارات PHP الحديثة قد يطلب وضع علامة ' او " حول كل فهرس حرفى .

ويمكن الوصول الى بيان السن مثلا هكذا:

```
$employee[age];
```

- استخدام الاقواس المربعة:

بنفس الطريقة يمكن تعريف المصفوفة بهذه الطريقة:

```
$employee[name] = "أيهاب";
$employee[occupation] = "محاسب";
$employee[age] = 20;
$employee["Department no"] = "5C";
```

المصفوفات المتعددة:

يمكن تعريف المصفوفات المتعددة بتعريف بسيط على انها مصفوفة داخل مصفوفة لذلك حتى تستطيع الوصول الى العنصر الثالث فى المصفوفة الثانية يجب ان تذكر فهرس كلا المصفوفتين هكذا:

```
myarray[1][2];
```

مثال:

```
<html>
<head>
<title>مصفوفة متعددة</title>
</head>
<body>
<?php
$Family = array (
array ( name=>"ايهاب",
occupation=>"محاسب",
age=>40,
depart=>"5C" ),
array ( name=>"جيهان",
occupation=>"طالبة",
age=>22,
Sex=>"انثى" ),
array ( name=>"اسامة",
occupation=>"مهندس",
age=>25,
Sex=>"ذكر" )
);

print $Family[0][occupation];
// النتيجة طالبة
?>
</body>
</html>
```

مما سبق يتضح كيف يمكن تعريف مصفوفة ثنائية البعد و الوصول الى العنصر الثانى من اول عائلة .

الطرق المختلفة للوصول الى المصفوفة:

هناك عدة طرق هامة يجب ان نعرفها لكى نستطيع الوصول الى المصفوفة وقراءة وتعديل قيمها المختلفة .

- الحصول على حجم المصفوفة:

يمكن الحصول على عدد العناصر فى المصفوفة عن طريق استخدام الدالة count كما يلى :

```
$item_color = array("احمر","اصفر" ,"اخضر" ,"ازرق" );
print $item_color[count($item_color)- 1];
```

مما سبق يتضح كيفية الوصول الى اخر عنصر فى المصفوفة لان الدالة count تحتوى على عدد العناصر بدء من الرقم واحد وبالتالى فان تنفيذ العبارة السابقة يؤدى الى طباعة اللون الازرق .

- التكرار داخل مصفوفة:

يمكن استخدام حلقة التكرار القوية foreach مع المصفوفة للوصول الى كل عناصرها هكذا :

```
foreach( $array as $temp )
{
```

```
//الكود الخاص بك هنا
}
```

حيث المعامل $array هو المصفوفة التى يتم التكرار داخلها و المعامل $temp هو المخزن الؤقت لكل عنصر داخل المصفوفة

مثال:

```
$item_color = array("احمر", "اصفر" ,"اخضر" ,"ازرق" );
foreach ($item_color as $c){
  print "$c<br>" ;
}
```

والان ماذا عن المصفوفات المترابطة ؟

يمكن التكرار داخل المصفوفة المترابطة عن طريق تعريف مفتاح العنصر وقيمة العنصر كما يلى:

```
foreach( $array as $key=>$value )
{
//الكود الخاص بك هنا
}
```

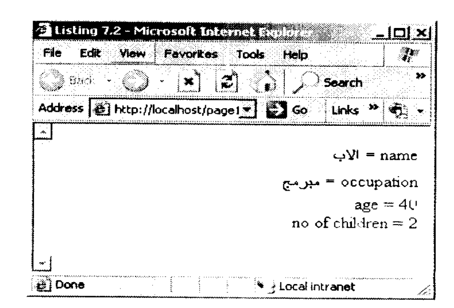
و يمكن اداء ذلك كما بالمثال الاتى:

```
1: <html dir="rtl">
2: <head>
3: <title>التكرار داخل المصفوفات</title>
4: </head>
```

```
<body>
<?php
$Family = array (
name=>"الاب",
occupation=>"مبرمج",
age=>40,
"no of children"=>"2"
);
foreach ( $Family as $key=>$val )
{
print "$key = $val<br>";
}

?>
</body>
</html>
```

مما سبق يتضح كيفية الوصول الى كل عنصر داخل المصفوفة المترابطة ومن المثال السابق يتم طباعة فهرس العنصر و قيمته كما بالشكل:



معالجة المصفوفات:

- دمج مصفوفتين:

يمكن دمج مصفوفتين و الوصول الى كل عناصرها عن طريق الدالة array_merge كمايلى

```
$first = array("a", "b", "c");
$second = array(1,2,3);
$third = array_merge( $first, $second );
foreach ( $third as $val )
{
print "$val<BR>";
}
```

نتيجة المثال السابق سيتم طبع قيم المصفوفتين اسفل بعضهم البعض

- أضافة عناصر الى المصفوفة:

يمكن أضافة المزيد من العناصر عن طريق الدالة array_push() التى تقوم بدمج عناصر جديدة فى المصفوفة الاصلية و تقوم ايضا بارجاع اجمالى عدد العناصر ى المصفوفة و يتضح ذلك من المثال:

```
$first = array("a", "b", "c");
$total = array_push( $first, 1, 2, 3 );
print " اجمالى عدد العناصر $total عنصر in \$first<P>";
foreach ( $first as $val )
{
print "$val<BR>";
}
```

فى المثال يتم طبع اجمالى عدد العناصر وهو 6 ثم يتم طبع عناصر المصفوفة الاصلية (a,b,c) ثم العناصر الجديدة (1,2,3)

لاحظ اننا قمنا بوضع علامة "\" قبل المتغير $first حتى يتم طبع حرف الدولار كما هو و لا يعتبره المعالج كمتغير ويقوم بطبع قيمته وهذا ماليس نريده.

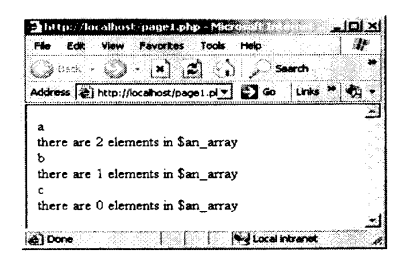
- حذف عناصر من مصفوفة:

تستطيع الدالة ()array_shift ان تقوم بحذف اول عنصر من المصفوفة مع ارجاعة و نستطيع عن طريق حلقة تكرار while ان نحذف جميع عناصر المصفوفة بحيث نختبر انتهاء العناصر من المصفوفة عن طريق الدالة count كمايلى:

```
<?php
  $an_array = array("a", "b", "c");
  while ( count( $an_array) )
  {
    $val = array_shift( $an_array);
    print "$val<BR>";
    print "there are ".count($an_array)." elements in
\$an_array <br>";
  }
?>
```

لاحظ ان الدالة count تصبح 0 عند حذف جميع العناصر من المصفوفة وهذا يعنى (false) بالنسبة لشرط الحلقة

وتكون نتيجة المثال السابق كما بالشكل:



-تقطيع المصفوفات:

يمكنك عن طريق الدالة array_slice() ان تقوم باستخراج عناصر معينة من مصفوفة الى مصفوفة جديدة وهذه الدالة تقبل ثلاث معاملات الاول هو المصفوفة التى تريد تقطيعها و الثانى وهو فهرس اول عنصر تريد البدء فى استخراجه (offset) و المعامل الاخير اختيارى وهو عدد العناصر التى تريد استخراجها واذا قمت بحذف هذا المعامل يتم استخراج جميع العناصر بدء من البداية المحددة وحتى النهاية و يتضح ذلك من المثال الاتى:

```
$first = array("a", "b", "c", "d", "e", "f");
$second = array_slice($first, 2, 3);
foreach ( $second as $var )
{
print "$var<br>";
}
```

هل تستطيع الان ان تستنتج نتيجة المثال السابق؟

يطبع المتصفح القيم c و d و e فقط لان المصفوفة تبدء من الصفر واذا قمت بتحديد رقم سالب فى معامل عدد العناصر فان جميع العناصر يتم استخراجها بدء من البداية المحددة .

– ترتيب المصفوفة:

من الاشياء الهامة جدا فى البرمجة عموما هو ترتيب البيانات حتى تكون نتيجة هذه البيانات عند طباعتها او عرضها على متخذى القرار مفهومة و منطقية و بدء من الاصدارة الرابعة للغة PHP تأتى اللغة بدوال قوية للترتيب نستعرضها فيما يلى:

الدالة Sort():

تقوم هذه الدالة بترتيب البيانات فى المصفوفة ابجديا اذا كانت تحتوى على نصوص او عدديا اذا كانت تحتوى على ارقام ولا تقوم هذه الدالة بارجاع اى بيانات عكس مثيلتها فى لغة Perl

```
$an_array = array("x","a","f","c");
sort( $an_array);
foreach ( $an_array as $var )
{
print "$var<BR>";
}
```

يمكنك ان تقوم بعكس عملية الترتيب (تنازليا) عن طريق الدالة rsort() بنفس الطريقة السابقة.

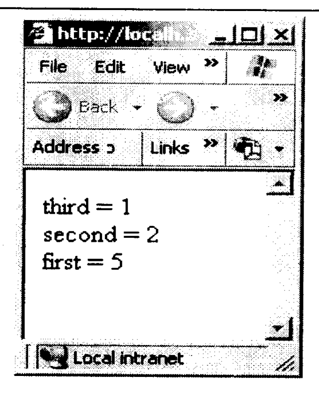
- فهرسة المصفوفات المترابطة:

لاتقوم باستخدام نفس الطريقة السابقة لترتيب البيانات مع المصفوفات المترابطة فسوف تقوم الدالة بالترتيب و لكن سيتم حذف مفتاح العناصر و استبدالها بارقام تناسب الترتيب الجديد و سنتعرف الان كيفية ترتيب المصفوفات المترابطة بدون حدوث ذلك.

لكى تستطيع ترتيب المصفوفات المترابطة يجب ان تقوم باستخدام الدالة asort() التى تستطيع ترتيب المصفوفة بدون حذف مفاتيح عناصرها لا المثال الاتى:

```
$first = array("first"=>5,"second"=>2,"third"=>1);
asort( $first );
foreach ( $first as $key => $val )
{
print "$key = $val<BR>";
}
```

وتكون نتيجة المثال الاتى كما بالشكل:



وبنفس الطريقة يمكنك عكس الترتيب باستخدام الدالة arsort()

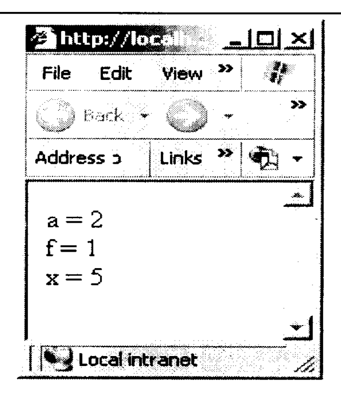
– ترتيب المصفوفات المترابطة عن طريق المفتاح:

يمكن عمل ذلك عن طريق الدالة ksort() التى تستقبل مصفوفة مترابطة ولا تقوم الدالة بارجاع اى قيمة .

مثال:

```
$first = array("x"=>5,"a"=>2,"f"=>1);
ksort( $first );
foreach ( $first as $key => $val )
{
print "$key = $val<BR>";
}
```

تكون نتيجة المثال السابق هو ترتيب العناصر بالشكل الاتى:


http://localh
File Edit View
Back
Address
Links
a = 2
f = 1
x = 5
Local intranet

الفصل الثامن

التعامل مع الكائنات المتجهة

التعامل مع الكائنات المتجهة:

من اهم المزايا فى لغة PHP هى امكانية استخدام الكائنات المتجهة و التى تتميز بانها لغة العصر الان لقوتها و ثباتها بالنسبة لانظمة التشغيل المختلفة و المفاهيم القادة تنطبق على كل اللغات الحديثة الموجودة الآن.

ماهو الكائن ؟

الكائن يمكنك ان تعتبره محتوى للدوال و المتغيرات المختلفة الذى ينحدر من نموذج خاص يسمى class و يقوم الكائن باخفاء عمله الداخلى عن الكود الذى يقوم باستخدامه وبدلا من ذلك يقدم واجهة سهلة للتعامل معه تسمى methods و التى بدورها لها وصول الى متغيرات خاصة تسمى properties .

تبدء البرمجة المتجهة بانشاء class او فئة لهذا الكائن و تتميز بمجموعة من السمات التى يتم توريثها للكائن بعد ذلك و لكن تختلف فى بعض خصائصها من كائن لاخر

و الفئة عبارة عن دوال خاصة تسمى methods و متغيرات خاصة تسمى properties

انشاء كائن:

لكى نستطيع انشاء كائن يجب ان ننشئ الفئة (class) المنحدر منها كما يلى:

```
class first_class
{
// وكود الفئة هنا
}
```

و يمكنك بعد ذلك ان تقوم باشاء نسخ من هذه الفئة كما يلى

```
$obj1 = new first_class();
$obj2 = new first_class();
print "\$obj1 is a ".gettype($obj1)."<br>";
print "\$obj2 is a ".gettype($obj2),"<br>";
```

لا حظ انه يجب ان تستخدم العبارة new() حتى يتم الانشاء وتقوم الدالة gettype() بارجاع نوع المتغير وفى حالتنا تقوم بارجاعة كلمة "object" التى يتم طباعتها

خصائص الكائنات:

يتم تعريف properties الخاصة بكائن عن طريق تعريف متغيرات خاصة توضع فى او تعريف الكائن كما يلى و يمكن ان تكون الخاصية عبارة عن قيمة حرفية او عددية او مصفوفة او حتى كائن اخر كما يتضح من المثال:

```
class first_class
{
var $name = "رامى";
}
```

لاحظ ان تعريف المتغيرات يتم عن طريق العبارة var والا سينتج خطأ

و الان تستطيع ان الوصول الى الخاصية name و حتى تغيير قيمتها الافتراضية كما يلى:

```
class first_class
{
var $name = "harry";
}

$obj1 = new first_class();
$obj2 = new first_class();
$obj1->name = "ماجد";
print "$obj1->name<BR>";
print "$obj2->name<BR>";
```

لاحظ انه تم تعديل الخاصية للكائن obj1 عن طريق العلامة <- التى يمكن عن طريقها الوصول الى الخصائص المتفرعة من الكائن

و يمكن استخدام الكائنات لتخزين البيانات المختلفة مثل المصفوفات ولكن بطريقة اكثر مرونة و اكثر قوة

وسائل الكائنات:

يتم توريث Object Methods من الفئات classes الابوية عند انشاء اى كائن بالعبارة ()new كما يتضح من المثال:

```
<html dir="rtl">
<head>
<title>Object Methods</title>
<body>
<?php
class first_class
{
var $name;
function sayHello()
{
print "مرحب بك";
}
}

$obj1 = new first_class();
$obj1->sayHello();
// يتم طبع كلمة مرحب بك
?>
</body>
</html>
```

لا حظ كيف تم الوصول الى الوسيلة التى.تم تعريفها فى داخل الفئة و بنفس الطريقة يمكن الوصول الى الخصائص:

```
<html dir="rtl">
<head>
<title>خصائص الكائنات</title>
<body>
<?php
```

```
class first_class
{
var $name="ماجد";
function sayHello()
{
print " أهلا بكم انا اسمى $this->name<BR>";
}
}

$obj1 = new first_class();
$obj1->sayHello();
// يتم طبع اهلا بكم انا اسمى ماجد //
?>
</body>
</html>
```

لاحظ استخدام العبارة this التى تشير الى الكائن الحالى و بنفس الطريقة يمكن الوصول الى المتغيرات و تغيير قيمتها داخل الوسائل المعرفة كما يلى:

```
<html dir="rtl">
<head>
<title>تغيير الخصائص</title>
</head>
<body>
<?php
131
class first_class
{
var $name="ماجد";
function setName( $n )
```

```
{
$this->name = $n;
}
function sayHello()
{
print " مرحب بك انا اسمى $this->name<BR>";
}
}


$obj1 = new first_class();
$obj1->setName("اسامة");
$obj1->sayHello();
// يتم طباعة مرحب بك انا اسمى اسامة //
?>
</body>
</html>
```

لاحظ انه تم تغيير قيمة الخاصية name و ان الكائن قام بالتحكم فى الخاصية عن طريق تقديم الوسيلة setName التى تستقبل معامل هو الاسم تماما مثل الدوال العادية.

هناك تقنية هامة تستخدم مع الفئات وهى امكانية انشاء وسيلة منشئة اى constructor method و يتم استدعائها تلقائيا عند انشاء الكائن اذا كان لها نفس اسم الفئة و يمكن للكائنات ان تقوم باستدعاء كود فى داخلها لكى تقوم بتمهيد نفسها كما يتضح من المثال الاتى:

```
<html dir="rtl">
<head>
<title>الوسائل المنشئة للكائنات</title>
</head>
<body>
<?php
class first_class
{
var $name;
function first_class( $n="ماجد" )
{
$this->name = $n;
}
function sayHello()
{
print " مرحب بك انا اسمى $this->name<BR>";
}
}
$obj1 = new first_class("اسامة");
$obj2 = new first_class("اشرف");
$obj1->sayHello();
// طباعة الرسالة مرحب بك انا اسمى اسامة
$obj2->sayHello();
// طباعة الرسالة مرحب بك انا اسمى اشرف
?>
</body>
</html>
```

عند انشاء الكائن يتم وضع القيمة الافتراضية "ماجد" فى خاصية الاسم فأذا لم نقم بتعيين قيمة للمعامل الخاص بالعبارة new() فأن قيمة هذه الخاصية تبقى كما هى

مثال عام:

سنقوم الان بتطبيق ما تعلمناه سابقا فى انشاء جدول يمكنه التعامل مع الحقول والصفوف بحيث يمكن التحكم فى عرض البيانات على شكل سطور اسفل بعضها

تعريف الخصائص:

اول خطوة هى معرفة ما هى الخصائص الهامة التى نحتاج الى تعريفها فسنحتاج فى هذا المثال الى تعريف مصفوفة للاعمدة و مصفوفة متعددة للصفوف و ايضا قيمة رقيمة لعدد الاعمدة التى نتعامل معها كما يلى:

```
class Table
{
var $table_array = array();
var $headers = array();
var $cols;
}
```

وسيلة الانشاء:

يمكن الحصول على عدد الاعمدة من constructor method بحيث يتم ارجاع اسماء الاعمدة فى مصفوفة وبالتالى يتم حساب عدد الاعمدة ووضعها فى خاصية جديدة cols :

```
function Table( $headers )
{
$this->headers = $headers;
$this->cols = count ( $headers );
}
```

وعن طريق تخزين هذه المعلومات فى خصائص الكائن فأن كل الوسائل Methods سيكون لها امكانية الوصول الى هذه المعلومات ايضا وسنبدء الان بانشاء هذه الوسائل:

الوسيلة ()addRow:

ستقوم هذه الوسيلة بزيادة عدد الصفوف التى هى على شكل مصفوفة مرتبة بنفس ترتيب الاعمدة وفيما يلى تعريف هذه الوسيلة

```
function addRow( $row )
{
 if ( count ($row) != $this->cols )
 return false;
 array_push($this->table_array, $row);
 return true;
}
```

تستقبل الوسيلة مصفوفة تساوى عناصرها عدد الاعمدة فى الجدول و يتم اختبار ذلك عن طريق الدالة count فأذا لم تكن متساوية فيتم ارجاع القيمة false.

تقوم الدالة array_push باضافة عناصر الى مصفوفة فأذا كان العنصر المضاف هو نفسه مصفوفة فيتم انشاء مصفوفة متعددة Multidimensional array.

أضافة الوسيلة ()addRowAssocArray:

تقدم هذه الوسيلة امكانية اضافة مصفوفة مترابطة بحيث لا يشترط اضافة القيم بنفس ترتيب الاعمدة و تقبل مفاتيح عناصر المصفوفة ويتم مقارنتها بالاعمدة فأذا لم تكن متماثلة يتم تجاهل هذه القيم و يتم تعريف هذه الوسيلة كما يلى:

```
function addRowAssocArray( $row_assoc )
{
$row = array();
foreach ( $this->headers as $header )
{
if ( ! isset( $row_assoc[$header] ))
$row_assoc[$header] = "";

$row[] = $row_assoc[$header];
}
array_push($this->table_array, $row);
return true;
}
```

يتضح من المثال كيفية اختبار مفاتيح العناصر للمصفوفة المضافة فأذا لم يتم وضع قيم لها فأن الدالة تقوم بوضع لا شئ "" فى هذا الحقل او العمود

لقد قمنا الان بتعريف وسيلتين للاضافة داخل المصفوفة و يتبقى وسيلة لعرض البيانات

الوسيلة ()Output:

تقوم هذه الوسيلة بعرض رأس الجدول و المصفوفة المخزنة فى خاصية array فى المتصفح ويتم تعريفها كمايلى:

```
function output()
{
print "<pre>";
foreach ( $this->headers as $header )
print "<B>$header</B> ";
print "\n";
foreach ( $this->table_array as $y )
{
foreach ( $y as $xcell )
print "$xcell ";
print "\n";
}
print "</pre>";
}
```

من الكود السابق يتضح انه يتم طباعة راس الجدول اولا ثم خلايا الجدول ولان هذه الخلايا عبارة عن مصفوفة ثنائية فنه يتم طباعة الجدول عن طريق حلقتين باستخدام العبارة foreach

وفيما يلى كود المثال بالكامل:

```
<html>
<head>
<title>مثال على الكائنات</title>
</head>
<body>
<?php
class Table
{
var $table_array = array();
var $headers = array();
var $cols;
function Table( $headers )
{
$this->headers = $headers;
$this->cols = count ( $headers );
}

function addRow( $row )
{
if ( count ($row) != $this->cols )
return false;
array_push($this->table_array, $row);
return true;
}

function addRowAssocArray( $row_assoc )
{
$row = array();
foreach ( $this->headers as $header )
{
if ( ! isset( $row_assoc[$header] ))
```

```
$row_assoc[$header] = "";
$row[] = $row_assoc[$header];
}
array_push($this->table_array, $row);
return true;
}

function output()
{
print "<pre>";
foreach ( $this->headers as $header )
print "<B>$header</B> ";
print "\n";
foreach ( $this->table_array as $y )
{
foreach ( $y as $xcell )
print "$xcell ";
print "\n";
}
print "</pre>";
}
}

$test = new table( array("a","b","c") );
$test->addRow( array(1,2,3) );
$test->addRow( array(4,5,6) );
$test->addRowAssocArray( array ( b=>0,
a=>6, c=>3 ) );
$test->output();
?>
</body>
</html>
```

وتكون نتيجة تنفيذ المثال السابق هو طبع الناتج:

```
a    b    c
1    2    3
4    5    6
6    0    3
```

مما سبق تتضح فوائد استخدام الفئات او class فى البرنامج فيمكنك اعادة استعمال الكود مرة اخرى فى أى مشروع يحتاج ان يقوم باظهار المعلومات بهذه الطريقة مما يوفر الوقت و الجهد.

الفصل التاسع

البيانات الزمنية

البيانات الزمنية

تقدم لغة PHP العديد من الادوات التى تمكننا من عمل حسابات زمنية لمختلف المجلات وفيما يلى اهم هذه الادوات.

الدالة ()Time:

تقوم الدالة ()time بارجاع التاريخ والوقت فى صورة رقم صحيح Integer لايعنى الكثير بالنسبة الينا فمثلا اذا كتبت الامر القادم:

- print time();

سيتم ارجاع رقم مثل هذا الرقم 1105177857 و هذا الرقم هو عبارة عن عدد الثوانى التى انقضت منذ منتصف الليل سنة 1/1/1970 وهو ما يسمى ببداية عصر اليونيكس .

ولا تندهش عندما تعرف انه يمكن استخراج جميع المعلومات المتعلقة بتاريخ و وقت معين من الرقم السابق فهناك ادوات اخرى تؤدى هذه الوظيفة و التعامل مع الزمن كرقم واحد يعطينا امكانية القيام بعمليات بحسابية معقدة مثلا لاضافة يوم واحد الى تاريخ و ارجاع الناتج فأذا لم تكن هذه الدوال موجودة فيجب ان تقوم بحساب نوع السنة (كبيسة / بسيطة) و ايام الشهور (31/30/29/28) و فيما يلى سنرى بساطة اداء ذلك

استخدام الدالة getdate():

الان لديك الرقم السابق وهو مايسمى timestamp و ستقوم الان بتحويل الرقم السابق الى تاريخ له معنى قبل ان تقوم بعرضة على الزائر و تقوم الدالة getdate() باستخدام معامل واحد قيمتة هى timestap الذى تريد تحويله و تقوم بارجاع مصفوفة مترابطة تحتوى على جميع البيانات الخاصة بهذا الزمن كما يتضح من الجدول الاتى :

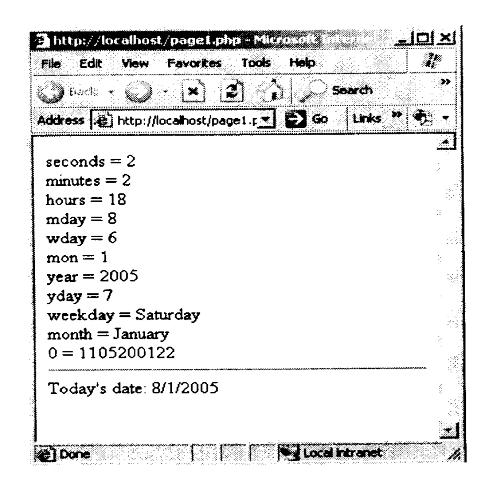
مثال	الشرح	المفتاح
28	عدد الثوانى من 0 الى 59	seconds
7	عدد الدقائق من 0 الى 59	minutes
12	الساعات من 0 الى 23	hours
20	أيام الشهر من 1 الى 31	mday
4	أيام الاسبوع من 0 الى 6	wday
1	شهور السنة من 1 الى 12	mon
2000	السنة مرقم ذو اربع حدود	year
19	أيام السنة من 0 الى 365	yday
Thursday	أسم اليوم	weekday

January	أسم الشهر	month
948370048	Timestamp الزمن	0

وبما ان المخرجات مصفوفة مترابط فيمكننا استخراجها كلها عن طريق الحلقة foreach كما يلى :

```
<html dir="rtl">
<head>
<title> الدالة getdate() </title>
</head>
<body>
<?php
$date_array = getdate(); // لا يتم وضع معامل للدالة لذلك
فــأن تاريخ اليوم سيتم استخدامه
foreach ( $date_array as $key => $val )
{
print "$key = $val<br>";
}
?>
<hr>
<?
print "Today's date:
$date_array[mday]/$date_array[mon]/
$date_array[year]<p>";
?>
</body>
</html>
```

وتكون نتيجة الكود السابق كمابالشكل:



يتضح من الشكل السابق كيفية اظهار عناصر الزمن المختلفة فى المصفوفة و يمكنك بالطبع الوصول الى اى عنصر لعرضه فى المتصفح لاحظ انك اذا قمت بتحديث الصفحة عن طريق الامر refresh (F5) سترى تغير ارقام الثوانى و ارقام عنصر timestamp كل ثانية .

الدالة Date():

تقتوم هذه الدالة بعرض التاريخ فقط اذا كنت تريد مجرد عرض التاريخ كنص حرفى وتقبل معامل حرفى هو التشكيل المراد اظهاره عند النداء على هذه الدالة و تقبل ايضا اختياريا المعامل timestamp.

مثال	الشرح	التشكيل
pm	للتعبير عن صباحا او مساء بحروف صغيرة	a
PM	للتعبير عن صباحا او مساء بحروف كبيرة	A
07	رقم اليوم بالنسبة للشهر مع صفر فى البداية	d
Thu	أسم اليوم بثلاثة حروف	D
January	أسم الشهر	F
12	الساعة بالارقام حتى 12 مع صفر فى البداية	h
12	الساعة بالنظام 24 مع صفر فى البداية	H
12	الساعات بالنظام 12 مع عد وجود صفر فى البداية	g
17	الساعة بالنظام 24 مع عدم وجود صفر فى البداية	G
47	الدقائق	i
4	رقم اليوم بالنسبة للشهر مع عد وجود صفر	j
Sunday	أسم اليوم بالكامل	1
1	السنة الكبيسة 1 اذا كانت كبيسة و 0 اذا لم تكن	L

01	رقم الشهر مع صفر فى البداية	m
Jun	اسم الشهر ثلاثة حروف فقط	M
6	رقم الشهر مع عدم وجود صفر فى البداية	n
24	الثوانى	s
948372444	الزمن timestamp	U
95	السنه حدين فقط	y
2005	السنة اربع حدود	Y
40	اليوم من 0 الى 365	z

مثال:

```
<html dir="rtl">
<head>
<title> الدالة date() </title>
</head>
<body>
<?php
print date("m/d/y G.i:s<br>", time());
// 1/9/2005 13.27:55
print "اليوم هو ";
print date("j of F Y, \a\\t g.i a", time());
//  2005 اليوم هو 9 من يناير
?>
```

```
</body>
</html>
```

الدالة mktime():

تقوم الدوال السابقة بارجاع الزمن الحالى ولكن ماذا اذا كنت تريد العمل مع زمن افتراضى اخر هنا يمكنك استخدام هذه الدالة التى تقوم بارجاع timestamp الذى يمكن وضعه كمعامل للدوال السابقة .

وتستقبل الدالة mktime() 6 معاملات عددية هم على الترتيب : الساعة – الدقائق – الثوانى – الشهر – اليوم – السنة .

مثال:

سنقوم فى هذا المثال باستخدام الدالة mktime() لتقوم بارجاع زمن لتاريخ معين يمكن استخدامه مع الدالة date() كمايلى:

```
<html dir="rtl">
<head>
<title> الدالة mktime</title>
</head>
<body>
<?php
// ارجاع الزمن للتاريخ 1/5/99 at 2.30 am
$ts = mktime( 2, 30, 0, 5, 1, 1999 );
print date("m/d/y G.i:s<br>", $ts);
// 05/01/99 2.30:00
```

```
print " التاريخ هو ";
print date("j of F Y, \a\\t g.i a", $ts );
// التاريخ هو 1 of May 1999, at 2.30 am
?>
</body>
</html>
```

وفى هذه الدالة يمكنك الاستغناء عن احد معاملاتها و سيتم استخدام الوقت او التاريخ الحالى و يمكنك ايضا زيادة عدد الساعات مثلا الساعة 25 ليتم اضافة ساعة على التاريخ المحدد.

أختبار تاريخ ()checkdate:

احيانا نحتاج ان نقوم باستقبال قيمة التاريخ من المستخدم فيمكننا التأكد عن طريق هذه الدالة من ان المستخدم قام بادخال التاريخ الصحيح بالنسبة للسنة المحددة.

مثال عام :

سنقوم الان بتجميع المعلومات السابقة لانشاء نتيجة يستطيع المستخدم من خلالها اختيار السنة و الشهر من خلال قوائم منسدلة بحيث تكون هذه النتيجة اولها سنة 1980 واخرها 2010 و سنقوم باستخدام المتغير العالمى $year و المتغير $month لنقوم بتسجيل اختيار المستخدم و اذا لم يقم المستخدم بكتابة اليوم سنأخذ اول يوم من ايام الشهر .

– سنقوم فى اول خطوة بفحص مدخلات المستخدم بحيث نتأكد من المتغيران $year و المتغير $month يحملان قيمة و يمكن التأكد من ذلك باستخدام الدالة ()isset ولكننا سنقوم فى هذا المثال باستخدام الدالة ()checkdate

كمايلى:

```
<?php
if ( ! checkdate( $month, 1, $year ) )
{
$nowArray = getdate();
$month = $nowArray[mon];
$year = $nowArray[year];
}
$start = mktime ( 0, 0, 0, $month, 1, $year );
$firstDayArray = getdate($start);
?>
```

الكود السابق هو جزء من المثال الكبير الذى سيتم تنفيذه ونقوم فيه اولا باختبار وجود قيم فى المتغييرين المستخدمين فأذا قامت الدالة checkdate() بارجاع القيمة false فيتم الحصول على التاريخ الحالى باستخدام الدالة getdate() التى تقوم بارجاع التاريخ الحالى فى مصفوفة مترابطة وبذلك يمكن استخدام الدالة mktime() و التى ستقوم بارجاع اول يوم فى الشهر و هو ماسنحتاج اليه لاحقا عن طري المتغير $firstdayarray

بناء نموذج الادخال:

سنقوم الان باستخدام الخاصية select حتى نستطيع ديناميكيا عرض الشهر الحالى او المختار .

```
<?php
if ( ! checkdate( $month, 1, $year ) )
```

```
{
$nowArray = getdate();
$month = $nowArray[mon];
$year = $nowArray[year];
}
$start = mktime ( 0, 0, 0, $month, 1, $year );
$firstDayArray = getdate($start);
?>
<html dir="rtl">
<head>
<title><?php print "النتيجة: $firstDayArray[month]
$firstDayArray[year]" ?></title>
<head>
<body>
<form method="post" action="<? print
$PHP_self ?>">
<select name="month">
<?php
$months = Array("يناير", "فبراير", "مارس", "ابريل",
"مايو", "يونيه", "يوليو", "أغسطس", "سبتمبر",
"أكتوبر", "نوفمبر", "ديسمبر");
for ( $x=1; $x <= count( $months ); $x++ )
{
print "\t<option value=\"$x\"";
print ($x == $month)?" SELECTED":"";
print ">".$months[$x- 1]."\n";
}
?>
</select>
<select name="year">
<?php
for ( $x=1980; $x<2010; $x++ )
```

```
{
print "\t<option";
print ($x == $year)?" SELECTED":"";
print ">$x\n";
}
?>
</select>
<input type="submit" value="أرسال">
</form>
</body>
</html>
```

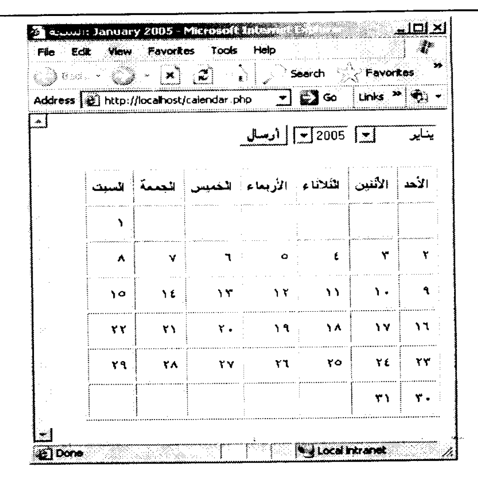
يجب بعد ان يختار المستخدم التاريخ ان نقوم بعرض ايام الشهر المختار وفيما يلي الكود الكامل لهذا المثال:

```
<?php
define("ADAY", (60*60*24) );
if (! checkdate( $month, 1, $year ) )
{
$nowArray = getdate();
$month = $nowArray[mon];
$year = $nowArray[year];
}
$start = mktime ( 0, 0, 0, $month, 1, $year );
$firstDayArray = getdate($start);
?>
<html dir= "rtl">
<head>
<title><?php print "النتيجة: $firstDayArray[month]
$firstDayArray[year]" ?></title>
<head>
```

```
<body>
<form action="<? print $PHP_SELF ?>"
method="post">
<select name="month">
<?php
$months = Array("يناير", "فبراير", "مارس", "ابريل",
"مايو", "يونيه", "يوليو", "أغسطس", "سبتمبر",
"أكتوبر", "نوفمبر", "ديسمبر");
for ( $x=1; $x <= count( $months ); $x++ )
{
print "\t<option value=\"$x\"";
print ($x == $month)?" SELECTED":"";
print ">".$months[$x- 1]."\n";
}
?>
</select>
<select name="year">
<?php
for ( $x=1980; $x<2010; $x++ )
{
print "\t<option";
print ($x == $year)?" SELECTED":"";
print ">$x\n";
}
?>
</select>
<input type="submit" value="أرسال">
</form>
<p>
<?php
$days = Array("الأحد", "الأثنين", "الثلاثاء", "الأربعاء",
```

```
"الخميس", "الجمعة", "السبت");
print "<TABLE BORDER = 1 CELLPADDING=5>\n";
foreach ( $days as $day )
print "\t<td><b>$day</b></td>\n";
for ( $count=0; $count < (6*7); $count++ )
{
$dayArray = getdate( $start );
if ( (($count) % 7) == 0 )
{
if ( $dayArray[mon] != $month )
break;
print "</tr><tr>\n";
}
if ( $count < $firstDayArray[wday] || $dayArray[mon] != $month )
{
print "\t<td><br></td>\n";
}
else
{
print "\t<td>$dayArray[mday] </td>\n";
$start += ADAY;
}
}
print "</tr></table>";
?>
</body>
</html>
```

وتكون النتيجة كما بالشكل :



الأحد	الأثنين	الثلاثاء	الأربعاء	الخميس	الجمعة	السبت
						١
٢	٣	٤	٥	٦	٧	٨
٩	١٠	١١	١٢	١٣	١٤	١٥
١٦	١٧	١٨	١٩	٢٠	٢١	٢٢
٢٣	٢٤	٢٥	٢٦	٢٧	٢٨	٢٩
٣٠	٣١					

ملاحظات:

- اخر حلقة For هى المسئولة فعليا عن طباعة أرقام الايام فى الجدول (أنظر الشكل) وهى بالطبع مرتبة .

- لقد قمنا بتمهيد متغير $count يحمل عدد مرات التكرار داخل الحلقة و قد حددنا ان تكون نهايته 42 (6*7) حتى تكون الخلايا المطبوعة كافية لكل ايام الشهر .

- فى السطر 54 تم اختبار قيمة المتغير $count اذا كانت ناتج بقية القسمة على 7 تساوى صفر هذا معناه ان الاختبار سيتحقق فقط اذا كان فقط المتغير

$count صفر او مضاعفات 7 وبهذه الطريقة يمكننا معرفة متى نقوم بانهاء التكرار او بداية صف جديد .

– فى السطر 56 قمنا بعمل اختبار للشهر الحالى $month مع اول يوم فى الشهر مخزن فى المتغير $dayarray[mon] و بالوصول الى بيان الشهر ومقارنته نعرف متى نصل الى نهاية الشهر وبذلك ننهى الحلقة .

– فى السطر 60 تم معرفة اول يوم من ايام الشهر بحيث يتم الطبع فى الخلية بالجدول عند اول يوم يوافق 1 من الشهر لذلك نقوم بمقارنة المتغير $count مع متغير يحمل اول يوم من الشهر $firstdayarray[wday] فأذا كان أقل منه نعرف اننا لايجب ان نقوم بطبع رقم اليوم فى هذه الخلية .

– فى السطر 64 العبارة else تتحقق عندما يكون المتغير $count يساوى المتغير $firstdayarray[wday] و نبدء فى طباعة ارقام الايام فى الخلية المناسبة من ايام الاسبوع عن طريق المصفوفة المترابطة $dayarray

– واخيرا نقوم بزيادة المتغير $start يوم كامل عن طريق اضافة عدد ثوانى يوم كامل (24*60*60) عن طريق المتغير الذى قمنا بتعريفه فى اول البرنامج ADAY.

الفصل العاشر

التعامل مع النصوص

التعامل مع النصوص

كل ماتراه امامك عند تصفح الانترنت من أشكل تفاعليه و قوائم رسومية هو فى الواقع عبارة عن نصوص بلغة HTML لذلك فالتعامل مع النصوص من الامور الهامة فى لغة PHP .

تشكيل النصوص:

حتى الان قمنا باستخدام العبارة print لطباعة مخرجات بسيطة و لكن تأتى الدالة ()printf و الدالة ()sprintf لتعطى اكثير من الامكانيات كما يتضح من الفقرات القادمة.

الدالة ()printf:

تستقبل الدالة عدة معاملات الاول دائما هو نص التشكيل و المعاملات الاخرى هى البيانات المراد طبعها فمثلا اذا اردت طباعة عدد و التعامل معه كرقم عشرى يمكن كتابة الامر:

```
printf("my number: %d",76);
```

و يتم طباعة الجملة:

```
my number: 76
```

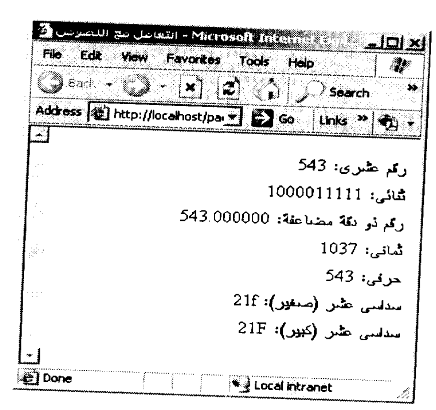
ونلاحظ استخدام علامة التشكيل المعرفة % والتى تأخذ أشكال عديدة نسردها فى الجدول الاتى :

الشرح	المعامل
يقوم بعض المعامل النصى كرقم عشرى	d
يقوم بعرض العدد الصحيح Integer كرقم ثنائى	b
طباعة الحرف المقابل لكود ascii	c
يقوم بعرض العدد الصحيح Integer كرقم ذو دقة مضاعفة Float	f
يقوم بعرض العدد الصحيح كرقم ثمانى (اساسه 8)	o
يقم بعرض المعامل (سواء ارقام او نصوص) كنص حرفى	s
يقوم بعرض العدد الصحيح Integer كرقم سداسى عشر باحرف صغيره (اساسه 16)	x
يقوم بعرض العدد الصحيح Integer كرقم سداسى عشر باحرف كبيرة	X

و لا نقوم بادراج المعاملات الخاص بتحويل النصوص (%) فقط و لكن يمكننا طباعة حتى وسوم HTML مثل
 كما يتضح من المثال :

```
1: <html dir="rtl">
2: <head>
```

```
<title>التعامل مع النصوص</title>
</head>
<body>
<?php
$number = 543;
printf("رقم عشرى: %d<br>", $number );
printf("ثنائى: %b<br>", $number );
printf("رقم ذو دقة مضاعفة: %f<br>", $number );
printf("ثمانى: %o<br>", $number );
printf("حرفى: %s<br>", $number );
printf("سداسى عشر (صغير): %x<br>", $number );
printf("سداسى عشر (كبير): %X<br>", $number );
?>
</body>
</html>
```



مثال 2:

فى صفحات الويب يتم التعامل مع الالوان كأرقام بالنظام السداسى عشر و يمكن تحويل الارقام العادية بالنظام RGB الى رقم يعبر عن نفس اللون بالنظام السداسى كما يلى:

```
$red = 204;
$green = 204;
$blue = 204;
printf( "#%X%X%X", $red, $green, $blue );
// يقوم بطباعة النص "#CCCCCC"
```

حشو النص:

احيانا نحتاج الى ان يتم طباعة الرقم دائما كحدين فمثلا اذا كان 5 يتم طباعته هكذا 05 و يمكن تنفيذ ذلك كمايلى:

```
printf( "%04d", 36 );
// يتم طباعة "0036"
```

ويمكنك ان تقوم بطبع الحشو كحرف اخر غير الصفر عن طريق الحرف ' كمايلى:

```
printf ( "%'x4d", 36 );
// يتم طباعة "xx36"
```

بالمثل أذا قمت بادخال المعاملات (1,1,1) فى المثال السابق الخاص بالالوان فأن القيمة 111 سيتم طباعتها و هذا خطأ لن يقبله المتصفح أذا حولت أستخدام هذا اللون فيجب كتابة عبارة printf() كمايلى:

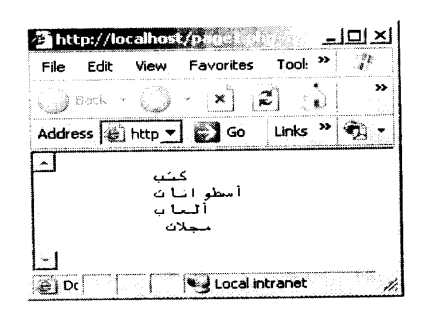
```
$red = 1;
$green = 1;
$blue = 1;
printf( "#%02X%02X%02X", $red, $green, $blue );
// يتم طباعة "#010101" //
```

أدراج مسافات:

يمكنك أدراج مسافات خالية فى النص المراد عرضه عن طريق حقل يسمى width او "عرض النص" ويتم تحديده بعد علامة "%" مباشرة و يمكن أداء ذلك كمايلى:

```
<html dir="rtl">
<?
print "<pre>";
printf("%20s\n", "كتب");
printf("%20s\n", "أسطوانات");
printf("%20s\n", "ألعاب");
printf("%20s\n", "مجلات");
print "</pre>";
?>
</html>
```

قمنا هنا بوضع الوسم <pre> حتى يتمكن المتصفح من عرض المسافات كما يتضح من الشكل:



أفتراضيا يتم وضع المسافات جهة اليمين و يمكن عكس ذلك الى جهة اليسار أذا قمت بوضع علامة – قبل الرقم هكذا .

```
printf("%-20s\n", "محازاة لليسار");
```

تحديد دقة الرقم:

يمكن للأرقام ذو الدقة المضاعفة أن تكون طويلة ولا يكون من المناسب عرض كل الرقم فى المتصفح لذلك يمكن تحديد الدقة لكى يتم تقريب الرقم الى حدود أقل كما يلى:

```
printf( "%.2f", 5.333333 );
// يتم طباعة الرقم "5.33"
```

لاحظ العلامة العشرية ثم رقم قبل العلامة "%" و يمكن تحديد حقل العرض فى عبارة واحدة هكذا "%4.2f" و لا يمكن استخدام معامل الحشو مع حقل العرض فى آن واحد و فيما يلى جدول يلخص الترتيب الذى يجب استخدامه عند كتابة نص التشكيل .

أسم التحويل	الشرح	مثال
معامل الحشو	يحدد عدد الحروف الواجب أدراجها	"4"
حقل العرض	يحدد عدد المسافات التى يجب أدراجها ضمن مخرجات النص	"20"
معامل الدقة	يحدد عدد الحدود الواجب التقريب اليها	".4"
معامل النوع	يحدد نوع البيانات التى يتم أخراجها	"d"

مثال:

سنقوم الان بعرض تقرير فى صفحة عبارة عن أسعار بعض المنتجات

```
1: <html dir="rtl">
2: <head>
3: <title> استخدام الدالة printf() لعرض أسعار المنتجات </title>
4: </head>
5: <body>
6: <?php
```

```
7: $products = Array( "كرسى سفره"=>"222.4",
8: "شمعة"=>"4",
9: "منضدة"=>80.6
10: );
11: print "<pre>";
12: printf("%-20s%23s\n", "أسم المنتج", "السعر");
13: printf("%'-43s\n", "");
14: foreach ( $products as $key=>$val )
15: printf( "%-20s%20.2f\n", $key, $val );
16: printf("</pre>");
17: ?>
18: </body>
19: </html>
```

قمنا فى هذا المثال أولا بتعريف مصفوفة مترابطة لعرض أسماء المنتجات و أسعارها و أيضا قمنا بتضمين الوسم <pre> حتى يتعرف المتصفح على المسافات التى نريد أظهارها فى الصفحة

فى السطر 12 أستخدمنا التشكيل :

"%-20s%23s\n"

ويعبر أول جزء منه "%-20s" عن تشكيل لبيانات حرفية بحيث يتم طباعة البيان فى عرض 20 حرف مع محازاة الى اليسار أما بقية التشكيل " %23s" يستخدم حقل عرضه 23 مع محازاة جهة اليمين و هذه العبارة هى المسئولة عن طباعة رأس التقرير .

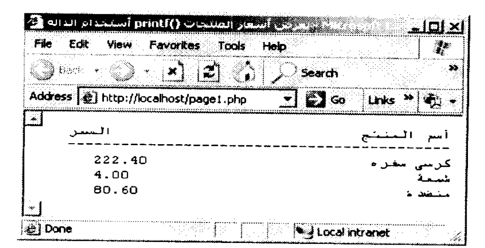
فى السطر 13 قمنا بطباعة سطر من الحرف "-" 43 مرة أسفل رأس البيان وتم ذلك عن طريق معامل الحشو الى يقوم بطباعة الحرف "-" الى نص فارغ .

```
printf("%'- 43s\n", "");
```

أخر عبارة printf() فى السطر 15 وهى تقع داخل الحلقة foreach التى تقوم بطبع محتويات المصفوفة .

```
printf( "%-20s%20.2f\n", $key, $val );
```

نلاحظ انه يتم طبع أسم المنتج فى حقل عرضه 20 مع محازاة الى اليسار ثم سعره فى حقل عرضه 20 مع دقه حدين و محازاة الى اليمين .



الدالة sprintf():

تعمل هذه الدالة مثل الدالة printf() تماما مع فرق واحد وهو انها لا تقوم بعرض المخرجات مباشرة على المتصفح بل تقوم بأرجاع القيمة الى متغير حرفى و يمكن بعد ذلك طبع محتويات هذا المتغير .

مثال:

```
$dosh = sprintf("%.2f", 2.334454);
print "أنت معك $dosh جنيه متاحة للصرف";
```

التعرف على النصوص:

تعطى لغة PHP العديد من الأدوات التى تمكننا من معرفة مصدر النص الذى نحصل عليه فقط يكوم من مدخلات مستخدم عن طريق نموذج أو من قاعدة بيانات او حتى من صفحة أخرى .

ملحوظة هامة:

النص الحرفى String هو عبارة عن مصفوفة تتكون من حروف فمثلا العبارة الاتية يمكن الوصول الى الحرف 3 و 8 كمايلى:

```
$test="النص الخاص بى";
print $test[2];
print $test[7];
```

معرفة طول النص:

من أهم المعلومات التى يحتاج اليها المبرمج كثيرا عند تعامله مع النصوص هو طول هذا النص ويمكن الوصول الى ذلك عن طريق الدالة ()strlen وهى دالة تقبل معامل حرفى و تقوم بأرجاع عدد صحيح هو عدد الحروف فى هذا النص.

ومن الأستخدامات النموذجية لهذه الدالة أختبار أدخال بيانات الى حقل معين من قبل المستخدم كما يمكن أيضاحه من المثال الأتى :

```
if ( strlen( $membership ) == 4 )
  print "مرحب بك";
else
  print "رقم الدخول يجب أن يتكون من 4 أرقام<P>";
```

البحث عن نص داخل نص:

يمكن الوصول الى أو البحث عن نص عبارة عن جزء من نص أكبر منه عن طريق الدالة strstr() التى تقبل معاملين هما النص الأساسى أو المصدر و النص المراد البحث عنه و تقوم الدالة بأرجاع true أذا وجدت النص أو false أذا لم تجده

مثال:

```
$membership = "pAB7";
if ( strstr( $membership, "AB" ) )
  print "مرحب بك – نرجو تذكر تجديد الأشتراك";
else
  print "مرحب بك";
```

من المثال السابق يتضح كيف تمكنا من طباعة نص مختلف بناء على قيمة فى متغير العضوية

الوصول الى نص داخل نص أخر :

بعكس الطريقة السابقة تقدم لنا الدالة ()strpos أمكانية الوصول الى فهرس (رقم الحرف فى المصفوفة الحرفية) أول حرف من النص المراد البحث عنه أذا وجده أما أذا لم يجده فيقوم بأرجاع القيمة false و تقبل الدالة معاملين النص المصدر و النص المراد البحث عنه و معامل ثالث أختيارى هو فهرس أول حرف المراد البدء بالبحث عنه .

```
$membership = "mz00xyz";
if ( strpos($membership, "mz") === 0 )
  print "hello mz";
```

لاحظ من المثال السابق أننا نريد البحث عن النص "mz" وهو ماستجده الدالة فى الموقع 0 أى أول حرف مما يؤدى الى تطوير عبارة IF الى false وهذا ما ليس نريده لذلك لكى نقوم بتفادى هذا الأجراء نقوم بأستخدام المعامل "===" الذى يتحقق فقط أذا كان طرفى التساوى متساويان فى القيمة و فى النوع أيضا

أستخراج جزء من نص:

تقوم الدالة ()substr بأستقبال معاملين هما النص الاصلى أو المصدر و الفهرس الذى يتم عنده البدء فى قطع النص من أوله كما يوجد معامل ثالث أذا تم تحديده يمكن التحكم فى عدد الحروف المقطوعة و هذا المعامل أختياريا

مثال:

```
$test = "نص حرفى";
print substr($test,6); // "فى" يتم طباعة
```

```
print substr($test,6,1) // "ف" يتم طباعة
```

أما أذا قمت بتحديد الفهرس الخاص ببداية القطع (المعامل الثانى) برقم سالب فأن بداية القطع تبداء تبدء من أخر النص بدلا من اوله

مثال:

```
$test = "embekheet@yahoo.com";
if ( $test = substr( $test, -9 ) == "yahoo.com" )
  print "أنت مستخدم بريد الكترونى فى الياهو";
else
  print "أنت لا تستخدم بريد الياهو";
```

النصوص المرمزة:

يمكن الحصول على أجزاء من نص واحد كبير عن طريق الدالة strtok() التى تقبل معاملين الأول هو النص الأصلى المراد تقطيعه و الثانى يسمى النص الفاصل و قد يكون أى عدد من الحروف و يتم أستخدام هذا النص الفاصل للفصل بين أجزاء النصوص وبعضها و حتى تتمكن من الوصول الى بقية الأجزاء يجب أن تنادى على هذه الدالة فى المرات التالية بدون تحديد النص الأصلى و لكن النص الفاصل فقط و يتم أستخدام هذه الدالة بطريقه نموذجيه داخل حلقة تكرار كما يتضح من المثال القادم .

مثال:

```
1: <html dir="rtl">
2: <head>
```

```
3: <title> الدالةstrtok()</title>
5: </head>
6: <body>
7: <?php
8: $test =
"http://www.egyptbooks.net/qs.xp?OP=dnquery.xp
&ST=MS&DBS=2&QRY=developer+php";
9: $delims = "?&";

10: $word = strtok( $test, $delims );
11: while (is_string( $word ) )
12: {
13: if ( $word )
14: print "$word<br>";
15: $word = strtok( $delims);
16: }
17: ?>
18: </body>
19: </html>
```

هذه الداله الى حد ما أمكانيتها ضعيفه لذلك علينا أن نقوم بأستخدامها بطريقه معينه لذلك قمنا أولا بتخزين النص الفاصل فى المتغير $delims

ثم نقوم بأستخدام الدالة strtok() أول مرة و تخزين الناتج فى المتغير $word السطر 10

ثم نقوم بأختبار قيمة المتغير $word داخل الحلقة while أذا كانت string فأذا لم تكن نعرف أنها نهاية النص و يجب الخروج من الحلقة

لاحظ أننا فى السطر 15 لا نقوم بتحديد النص الأصلى فقط النص الفاصل و ألا سندخل الى حلقة تكرارية لانهائية

الدالة ()trim و ()ltrim:

عندما تقوم بأستقبال قيمة نصية من مستخدم فأنت لا تعرف أذا كان النص يوجد به مسافات خالية أم لا لذلك يجب أستخدام الدالة ()trim التى تقوم بحذف المسافات الخالية و الtabs من أول و أخر النص و تقوم هذه الدالة بأستقبال النص المراد تعديله و تقوم بأرجاع النسخة الجديدة منه .

مثال:

```
$text = "\t\t\tنص حرفى به  مسافات كثيرة       ";
$text = trim( $text );
print $text;
// "نص حرفى به  مسافات كثيرة" يتم طباعة;
```

أما أذا كنت تريد الأحتفاظ بالمسافات فى أول النص و ألغاؤها فى أخر النص فيمكنك أستعمال الدالة ()chop تماما مثل الدالة ()trim

مثال:

```
$text = "\t\t\tنص حرفى به  مسافات كثيرة       ";
$text = chop( $text );
print $text;
```

```
;"نص حرفى به   مسافات كثيرة            " يتم طباعة //
```

و عكس عمل الدالة chop() فأن الدالة ltrim() تقوم بالأحتفاظ بالمسافات الخالية فى أخر النص و تحذفها فى اخره كما يتضح من المثال

مثال:

```
$text = "\t\t\tنص حرفى به  مسافات كثيرة      ";
$text = ltrim($text);
print $text;
// prints "نص حرفى به  مسافات كثيرة      "; //
```

أستبدال نص الدالة substr_replace():

تشابه هذه الدالة عمل الدالة substr() ولكنها تقوم بتعديل النص التى تقوم بأستخراجه و تقبل هذه الدالة ثلاثة معاملات الأول هو النص المراد تعديله و الثانى هو النص المراد أضافته و الأخير هو الفهرس الذى تبدء عنده عملية الأستبدال و تستطيع أن تدرج معامل رابع أختيارى وهو طول النص المراد أستبداله

و بذلك تستطيع هذه الدالة أن تقوم بأستبدال النص الأصلى بالنص المعدل بدء من الحرف الذى يحدده معامل الفهرس و نهاية الى أخر النص أو الى عدد الحروف معين أذا تم تحديد المعامل الرابع

مثال:

فى هذا المثال نفترض أن المستخدم عضويته أنتهت لانه يوجد الرقمين 99 فى كود العضوية الخاص به و نريد تجديد العضوية بالنسبة له الى 00 فنستخدم الدالة كمايلى

```
<?
$membership = "mz99xyz";
$membership = substr_replace($membership, "00",
2, 2 );
print "رقم العضوية الجديد: $membership<p>";
?>
```

أستبدال النص بأستخدام الدالة :str_replace()

تقبل هذه الداله ثلاثة معاملات الأول هو النص المراد البحث عنه لأستبداله والثانى هو النص الجديد الذى سيتم أستخدامه و الأخير وهو النص الأصلى الذى ستم عمل التعديلات به .

مثال:

سنقوم فى النص القادم بتغيير النص"05" الى "2005"

```
$string = "05 جميع الحقوق محفوظة لسنة. ";
$string .= "05 الموقع الجديد لدار البراء سنة";
print str_replace("05","2005",$string);
```

لاحظ ان المتغير $string يوجد بعده علامة النقطة التى تقوم بأضافة النص الجديد اليه و عدم ألغاء القديم.

تحويل حالة الأحرف:

عندما تريد مقارنة نص يتم أدخاله بواسطة مستخدم و تريد مقارنته بقيمة حرفية فأنت لا تعرف اذا كان المستخدم سيقوم بأدخال حروف كبيرة أو صغيرة لذلك حتى تتم المقارنة بطريقة صحيحة يجب تحويل النصين المراد مقارنتهم الى حروف صغيرة أو حروف كبيرة.

و تستخدم الدالة strtoupper() لتقوم بتحويل النص الى حروف كبيرة و تقبل معامل واحد وهو النص المراد تعديله و ترجع النص المعدل

مثال:

```
$membership = "mz00xyz";
$membership = strtoupper($membership);
print "$membership<P>"; // prints "MZ00XYZ"
```

وبالمثل يمكن تحويل النص الى حروف صغيرة بأستخدام الدالة strtolower() كما يلى:

```
$my_addrress = "WWW.EGYPTBOOKS.NET";
$my_addrress = strtolower( $my_addrress );
print "$my_addrress <P>"; // تطبع
"www.egyptbooks.net"
```

هناك دالة أخرى مفيده و هى الدالة ()ucwords التى تقوم بتحويل أول حرف من كل كلمة الى حرف كبير .

```
$my_msg = "windows control panel";
$my_msg = ucwords( $my_msg);
print "$my_msg <P>"; // يطبع "Windows Control
Panel"
```

أذا قمت بكتابة نص على الشكل "winDOws conTRoL paNeL" فأن الدالة السابقة بمفردها لن تحسن شكل النص و لكن يجب أستخدام الدالة
()strtolower أولا لتحويل كل الحروف الى حروف صغيرة ثم نقوم بأستخدام الدالة ()ucwords

تقسيم النص الى مصفوفة:

يمكنك تقسيم النص الى مصفوفة حرفية عن طريق الدالة ()explode و هى مشابه لعمل الدالة ()strtok فهى تقبل معاملين الأول هو النص الفاصل و الثانى هو النص الأصلى الذى تريد تقسيمه و تقوم الدالة بأرجاع مصفوفة حرفية و يمكنك أستعمال اى عدد من الحروف كنص فاصل فهذه الدالة تتعامل معهم كلهم كوحدة واحدة عكس النص الفاصل فى الدالة ()strtok فأن كل حرف يحدد كنص فاصل يعامل كوحدة منفصلة

مثال:

تقسيم التاريخ الى عدة حقول

```
$start_date = "1976-06-22";
$date_array = explode("-", $start_date);
// $date[0] == "1976"
// $date[1] == "06"
// $date[2] == "22"
```

الفصل الحادى عشر

التعامل مع الملفات

التعامل مع الملفات

تعطى لغة PHP الكثير من الأدوات التى تمكنك من التعامل مع الملفات مثل القراءة و الكتابة

الدالة ()Include:

تستخدم هذه الدالة لتضمين ملف خارجى الى الملف الحالى فيصبح الملف الخارجى كأنه جزء من الملف الحالى و هذا مفيد حيث أنك تستطيع تضمين ملف به العديد من الدوال الجاهزة الأستخدام (مكتبه) الى الملف الحالى مما يعطى أمكانيه سهلة و سريعة للوصول الى هذه الدوال من اى صفحة تستخدم هذه المكتبه .

و أذا أردت أن تقوم بتطوير المكتبه أو تصليح خطأ أكتشفته فيها فليس عليك الا التعديل فى هذه المكتبه فقط و ليس فى كل الصفحات التى تستخدمها

مثال:

- قم بأنشا ملف أسمه myfunc.php و به الكود الأتى:

```
<?php
$retval = ( 4 + 4 );
return $retval;
?>
```

- قم الأن بأنشاء ملف أخر بأى أسم ثن أكتب الكود الأتى به

```
<html dir="rtl">
<head>
<title>مثال على الدالة تضمين الملفات</title>
</head>
<body>
<?php
$addResult = include("myfunc.php");
print "الملف الخارجى قام بأرجاع القيمة $addResult";
?>
</body>
</html>
```

لاحظ أن الدالة include() قامت بإرجاع قيمة العملية الحسابية الى المتغير $addresult الى قمنا بعد ذلك بطباعة قيمته

يمكننا أيضا تضمين ملفات خارجيه بناء على تحقق شرط معين و يمكن أن يتم ذلك بالطريقه الأتيه

```
$test = false;
if ( $test )
{
include( "file.txt" );
}
```

لاحظ أن العبارة If لا يتم تحققها و بالتالى فأن الملف "file.txt" لن يتم تضمينه أبدا

وإذا قمت بتضمين الدالة ()include فى حلقة تكراريه فأن محتويات الملف الخارجى يتم تنفيذها فى كل مرة يتم فيها النداء على الدالة و فيما يلى مثال يقوم بالنداء على ثلاثة ملفات مختلفة .

```
<html dir="rtl">
<head>
<title>تنفيذ الملفات الخارجية داخل حلقة </title>
</head>
<body>
<?php
for ( $x = 1; $x<=3; $x++ )
{
$incfile = "incfile$x".".txt";
print "بداية تنفيذ الملف $incfile<br>";
include( "$incfile" );
print "<p>";
}
?>
</body>
</html>
```

يتم فى المثال السابق تنفيذ الحلقة ثلاثة مرات و يتم أيضا النداء على ثلاثة ملفات مختلفة هم "incfile1.txt" و "incfile2.txt" و "incfile3.txt"

يوجد منذ الأصدارة 4 من لغة PHP العبارة ()require التى يمكنها عمل نفس وظيفة الدالة ()include و لكن بدون أرجاع قيمة

الدالة ()file_exists:

عادة قبل التعامل مع الملفات الخارجية سواء بالقراءة أو الكتابة فأن أختبار وجود الملف من الاشياء الهامة حتى لا ينتج خطأ وقت التشغيل و تستخدم الدالة ()file_exists لهذا الغرض فهى تستقبل معامل واحد وهو مسار الملف المطلق او النسبى و ترجع أما القيمة true أذا وجدت الملف أو القيمة false أذا لم تجده

مثال:

```
if ( file_exists("test.txt") )
  print "الملف موجود";
```

يمكنك أيضا أختبار وجود ملف معين عن طريق الدالة ()is_file التى ترجع قيمة من نوع boolean

```
if (is_file("test.txt"))
  print "test.txt  ملف";
```

نفس الطريقه يمكن أختبار وجود دليل فرعى عن طريق الدالى ()is_dir التى ترجع أيضا قيمة boolean

```
if (is_dir("/php"))
  print "php  دليل فرعى";
```

هناك دوال أخرى تمكنك من التعامل مع الملفات بأحترافية فهى تعبر عن حالة الملفات الموجودة على السيرفرات المختلفة التى يوجد بها نظام آمان معين قد

يمنعك من التعامل مع الملف وفيما يلى الدوال التى تمكنك من معرفة صلاحياتك بالنسبة لملف معين

الدالة is_readable():

تمكنك الدالة من معرفة أذا كنت تستطيع قراءة الملف أم لا (أى تستطيع فتحه أم لا) و فى سيرفرات يونيكس قد تستطيع رؤية الملفات الملف و لا تستطيع فتحه و يمكنك تنفيذ هذه الدالة كما يلى :

```
if ( is_readable( "test.txt" ) )
  print " يمكنك فتح وقراءة الملف";
```

بالمثل يمكنك تحديد أمكانية التعديل فى ملف عن طريق الدالة is_writable()

```
if (is_writable( "test.txt" ) )
  print " يمكنك فتح وقراءة الملف";
```

وأيضا فى أذا كنت تريد تنفيذ ملف معين و بناء على أمتداد الملف (نوعه) وصلاحيات الدخول الخاصة بك فى السيرفر يتم التحقق من ذلك عن طريق الدالة is_executable() كما يلى :

```
if ( is_executable( "test.txt" )
print "الملف قابل للتنفيذ";
```

حجم الملف:

يمكنك معرفة حجم ملف معين عن طريق الدالة ()filesize التى تقبل معامل واحد هو مسار الملف و ترجع حجمه بالبايت .

```
print "حجم ملف قاعدة البيانات: "
print filesize("mydb.mdb");
```

الحصول على تاريخ الملف:

أحيانا قد تريد معرفة متى أخر مرة تم فيها فتح الملف أو تعديله و تقدم لغة PHP العديد من الدوال التى تمكنك من معرفة ذلك

فأذا أردت الحصول على تاريخ أخر وصول الى الملف يمكنك النداء على الدالة ()filetime التى تقوم بأرجاع التاريخ فى صورة رقمية و هى عدد الثوانى منذ 1 يناير 1970 (عصر اليونيكس) و يمكن أستخدام الدالة ()date حتى يمكن تحويل الرقم الناتج الى تاريخ قابل للفهم

```
$atime = fileatime( "test.txt" );
print "أخر مرة تم فيها الوصول الى الملف ";
print date("D d M Y g:i A", $atime);
// سيتم طباعة تاريخ مثل هذا: Thu 17 Jan 2001 2:26 PM
```

راجع فصل التعامل مع البيانات الزمنية

ويمكن أيضا معرفة متى أخر مرة تم فيها تعديل الملف أو تغيير صلاحياته عن طريق الدالة filectime()

```
$ctime = filectime( "test.txt" );
print " تم تعديل الملف أخر مره فى ";
print date("D d M Y g:i A", $ctime);
// سيتم طباعة تاريخ مثل هذا: Thu 17 Jan 2001 2:26 PM
```

مثال عام:

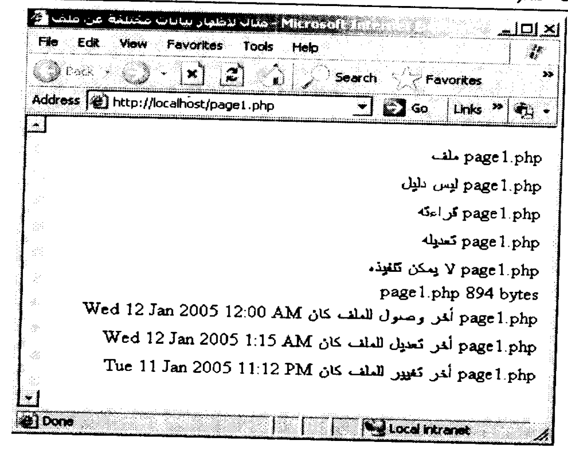
سنقوم الأن ببناء دالة تقوم بأرجاع بيانات مختلفة عن ملف معين بأستخدام المهارات السابقة

```
<html dir="rtl">
<head>
<title>مثال لاظهار بيانات مختلفة عن ملف</title>
</head>
<body>
<?php
$file = "page1.php";
outputFileTestInfo( $file );
function outputFileTestInfo( $f )
{
if ( ! file_exists( $f ) )
{
print "$f الملف غير موجود<BR>";
return;
}
```

```
print "$f ".(is_file( $f )?"":" ليس")."ملف<br>";
print "$f ".(is_dir( $f )?"":" ليس")."دليل<br>";
print "$f ".(is_readable( $f )?"":" لا يمكن ")."قراءته<br>";
print "$f ".(is_writable( $f )?"":" لا يمكن ")."تعديله<br>";
print "$f ".(is_executable( $f )?"":" لا يمكن ")."تنفيذه<br>";
print "$f ".(filesize($f))." bytes<br>";
print "$f أخر وصول للملف كان ".date( "D d M Y g:i A", fileatime( $f ) )."<br>";
print "$f أخر تعديل للملف كان ".date( "D d M Y g:i A", filemtime( $f ) )."<br>";
print "$f أخر تغيير للملف كان ".date( "D d M Y g:i A", filectime( $f ) )."<br>";
}

?>
</body>
</html>
```

و يكون تنفيذ المثال يشابه الشكل القادم :



لاحظ انه تم أستخدام المعامل الثلاثى الذى يمكن عن طريقه أختبار بعض الحالات فى سطر واحد بدون كتابة الكثير من عبارات IF و يمكن تنفيذ هذا المعامل كما يلى:

```
"$f ".(is_file( $f )?"":"ليس ")."ملف<br>";
```

لاحظ أن العلامة ? تغنى عن العبارة IF فأذا تحقق الشرط تم تنفيذ الجزء الذى يأتى بعدها و هو طبع نص فارغ "" أما أذا لم يتحقق الشرط فيتم طبع العبارة "ليس" و يتم وصل كل النصوص الحرفية عن طريق معامل الربط ".".

ويمكن تنفيذ نفس الكود السابق بالطريقه الأتيه :

```
if ( is_file( $f ) )
print "$fi ملف<br>";
else
print "$fi ليس ملف<br>";
```

حذف و أنشاء الملفات:

أذا لم تجد الملف بأستخدام الدالة ()file_exists فيمكنك أنشاؤه بأستخدام الدالة ()touch التى تقبل معامل واحد هو مسار الملف و أسمه و تقوم بأنشاؤه فارغا أما أذا كان الملف موجودا من قب فأن تنفيذ هذه الدالة لا يقوم بإلغاء الملف القديم و لكن يتم تعديل تاريخ أخر وصول فقط

```
touch("file.txt");
```

و بنفس الطريقة يمكنك حذف ملف عن طريق الدالة ()unlink التى تقبل مسار الملف الذى تريد حذفه

لاحظ أنه يجب وضع صلاحيات معينه للملف الذى تريد العمل عليه

فتح الملف للكتابة أو القراءة:

قبل أن تقوم بالتعامل مع الملف فيجب أن تقوم بفتحه أولا و يتم ذلك عن طريق الدالة ()fopen التى تقبل معاملين الأول هو مسار الملف المراد فتحه و الثانى هو الحالة أو نوع الفتح (mode) .

نوع الفتح قد يكون قراءة (r) أو كتابة (w) او للأضافه (a) وتقوم الدالة بعد الفتح بإرجاع عدد صحيح Integer هو يعبر عن مؤشر فى الذاكرة لهذا الملف و يمكن عن طريقه الوصول الى الملف و يسمى (file pointer) و يجب تخزين هذه القيمه فى متغير

مثال:

```
$fp = fopen( "test.txt", 'r' );
```

العبارة السابقة تقوم بفتح الملف "test.txt" للقراءة فقط و يتم تخزين مؤشر الملف فى المتغير fb

مثال:

أيضا الدالة ()fopen تقوم بإرجاع القيمة false أذا حدثت أى مشكله تعوق عملية الفتح و يمكن أختبارها مثلا قبل الكتابه فى ملف

```
if ($fp = fopen( "test.txt", 'w' )) {
// كود الكتابة
}
```

و يمكن تنفيذ المثال السابق بطريقه مشابه فى سطر واحد هكذا

```
($fp = fopen( "test.txt", "w" ) ) or die(" لا يمكن فتح
الملف");
```

إذا تم فتح الملف بطريقه صحيحه فإن الجزء الثانى من العبارة بعد or لا يتم تنفيذه أما أذا حدثت مشكله في فتح الملف فأن الدالة die() يتم تنفيذها و التى تقوم بطبع رساله فى المتصفح و أيضا أنهاء الكود الحالى .

كخطوة أخيرة فى التعامل مع الملف يجب عليك عند الأنتهاء من القراءة أو الكتابة أن تقوم بإغلاق الملف و تقوم الدالة fclose() بهذه الوظيفه و التى تقبل معامل واحد هو مؤشر الملف الذى تم ارجاعه من تنفيذ صحيح للداله fopen()

```
fclose($fp);
```

القراءة من الملفات:

تعطى لك لغة PHP أكثر من طريقه للقراءة من الملفات فتستطيع قراءة الملف بالحرف أو بالبايت أو بالسطر .

– القراءة بالسطر تتم عن طريق الدالة fgets() التى تقبل معاملين الأول هو مؤشر الملف الذى تم إرجاعه من الدالة fopen() و الثانى عدد صحيح يعبر عن عدد البايت التى يجب للدالة قراءته أذا لم تصل الى نهاية السطر أو الى نهاية الملف .

وهكذا نستطيع أن نستنتج أن الدالة fgets() تستمر فى القراءة حتى تصل الى نهاية السطر و تجد الحرف "\n" أو الى نهاية الملف أو الى عدد البايت المحدد كمعامل ثانى لها .

```
$line = fgets( $fp, 1024 );
```

و رغم أن الدالة ()fgets تنتهى عند نهاية الملف فأنك يجب أن تعرف متى يتم أيقاف القراءة عند نهاية الملف و تقوم الدالة ()feof بهذه الوظيفه و التى تقبل معامل واحد هو مؤشر الملف وتقوم بأرجاع القيمة true عند الوصول الى نهاية الملف .

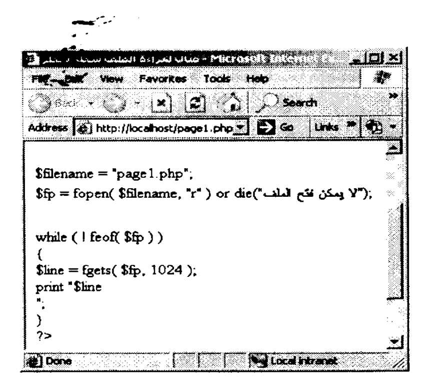
مثال:

```
1: <html>
2: <head>
3: <title>مثال لقراءة الملف سطر سطر</title>
4: </head>
5: <body>
6: <?php
7: $filename = "page1.php";
8: $fp = fopen( $filename, "r" ) or die(" لا يمكن فتح الملف");
9: while ( ! feof( $fp ) )
10: {
11: $line = fgets( $fp, 1024 );
12: print "$line<br>";
13: }
14: ?>
15: </body>
16: </html>
```

لاحظ فى السطر 8 أن العبارة بعد or يتم تنفيذها عند وجود مشكله فى فتح الملف و بالتالى يتم أنهاء باقى الكود بالكامل وهذا عادة يحدث فى حالة عدم وجود الملف أو عد وجود صلاحيات للقراءة من الملف فى النظام unix مثلا

فى السطر 9 يوجد الشرط الخاص بالقراءة من الملف حتى نهايته فالمعامل ! لو تتذكر يعبر عن النفى و الشرط فى هذا السطر معناه أنه أذا لم تصل الى نهاية الملف فأستمر فى تنفيذ الحلقة و بالتالى القراءة من الملف .

و يتم طباعة محتويات الملف سطر بسطر فى السطر 12 الذى يقوم بطباعة المتغير $line ووضع الوسم
 حتى تتم الطباعة أسفل بعضها البعض و يمكن قراءتها بسهولة .



لاحظ كيف تمكنت من قراءة ملف الكود الأصلى ("page1.php") و عرضه فى المتصفح بهذه الطريقه

الدالة fread():

بدلا من قراءة الملف سطر بسطر يمكنك قراءة مقدار معين من البيانات دفعة واحدة عن طريق هذه الدالة التى تقبل معاملين الأول هو مؤشر الملف و الثانى

مقدار البيانات التى تريد قراءتها بالبايت و تقوم الدالة بإرجاع البيانات التى طلبتها أو حتى تصل الى نهاية الملف .

```
$chunk = fread( $fp, 16 );
```

مثال:

سنقوم بتنفيذ نفس المثال السابق لكننا سنقوم بقراءة 24 بايت من محتويات الملف فى كل مرة بدلا من سطر سطر

```
<html dir="rtl">
<head>
<title> مثال على الدالة fread()</title>
</head>
<body>
<?php
$filename = "test.txt";
$fp = fopen( $filename, "r" ) or die(" لا يمكن فتح
الملف");
while ( ! feof( $fp ) )
{
$chunk = fread( $fp, 24 );
print "$chunk<br>";
}
?>
</body>
</html>
```

و لا تعطيك الدالة ()fread أمكانية تحديد بداية موضع القراءة و لكن يمكنك تحقيق ذلك عن طريق الدالة ()fseek التى تقبل معاملين الأول هو مؤشر الملف و الثانى هو رقم الموضع الذى تريد الأنتقال اليه بالبايت .

```
fseek($fp, 43);
```

مثال:

```
<html>
<head>
<title> مثال على الدالة fseek()</title>
</head>
<body>
<?php
$filename = "myfile.txt";
$fp = fopen( $filename, "r" ) or die("لا يمكن فتح الملف");
$fsize = filesize($filename);
$halfway = (int)( $fsize / 2 );
print " يوجد منتصف الملف عند: $halfway <BR>\n";
fseek( $fp, $halfway );
$chunk = fread( $fp, ($fsize - $halfway) );
print $chunk;
?>
</body>
</html>
```

هل تستطيع أن تستنتج عمل المثال السابق ؟

يقوم المثال السابق ببساطة بطباعة نصف الملف "myfile.txt" و أستطعنا أتمام ذلك عن طريق الدالة filesize() و قمنا بقسمة الناتج على 2 فى السطر 10

ثم قمنا بالأنتقال الى نقطة المتصف عن طريق الدالة fseek() فى السطر 12 و قمنا بتخصيص باقى المساحة المتبقية من الملف الى الدالة fread() حتى يتم قراءة باقى الملف دفعة واحدة و أخير ا تم طباعة الملف فى السطر 14

الدالة fgetc():

وهذه الدالة تشبه الى حد كبير الدالة fgets() و لكنها تقوم بإرجاع حرف واحد فى كل مرة يتم فيها النداء على هذه الدالة و نظرا لان حرف واحد يمثل بايت واحد دائما فأن معامل طول القراءة غير موجود فى هذه الدالة

```
$char = fgetc( $fp );
```

مثال:

```
<html dir="rtl">
<head>
<title> مثال على الدالة fgetc()</title>
</head>
<body>
<?php
$filename = "myfile.txt";
$fp = fopen( $filename, "r" ) or die(" لا يمكن فتح
الملف");
while (! feof($fp))
```

```
{
$char = fgetc( $fp );
print "$char<BR>";
}
?>
</body>
</html>
```

مثل ماسبق تماما مع الدالة fgets() يتم القراءة من الملف داخل حلقة تكرارية حتى نهاية الملف و لكن تتم القراءة حرف واحد فى كل مرة

تعديل الملفات:

تعديل الملف لا يختلف كثيرا عن فتحه للقراءة فيمكنك أن تقوم بتغيير المعامل الثانى للداله fopen() من "r" الى "w" للكتابة

```
$fp = fopen( "myfile.txt", "w" );
```

أذا كان الملف غير موجودا فيتم أنشاؤه و أذا كان موجودا فيتم تدمير البيانات القديمه و أستبدالها بالبيانات الجديده عن البدء فى الكتابه

و بالمثل يمكنك تغيير المعامل الثانى للداله من "r" الى "a" للأضافه

```
$fp = fopen( "myfile.txt", "a" );
```

و تتم الأضافه الى البيانات الموجودة و لا يتم حذفها

الكتابه الى الملف:

يمكنك الكتابة الى الملف عن طريق الدالة fwrite() او الدالة fputs() و هما يعملان بنفس الطريقة تماما فيتم تحديد معاملين الأول هو مؤشر الملف و الثانى هو النص الحرفى المراد كتابته .

```
fwrite( $fp, "مرحب بك" );
fputs( $fp, "مرحب بك" );
```

مثال:

```
<html dir="rtl">
<head>
<title>مثال على الكتابة داخل الملفات</title>
</head>
<body>
<?php
$filename = "myfile.txt";
print "سيتم الكتابة الى الملف $filename<br>";
$fp = fopen( $filename, "w" ) or die(" لا يمكن فتح الملف ");
fwrite( $fp, "مرحب بك\n" );
fclose( $fp );
print " الأضافة الى الملف $filename<br>";
$fp = fopen( $filename, "a" ) or die(" لا يمكن فتح الملف");
fputs( $fp, "أضافة نص أخر الى الملف\n" );
```

```
fclose( $fp );
?>
</body>
</html>
```

أقفال الملف:

فى الحياة العملية الكثير من المستخدمين أو الزائرين لموقعك سيقوموا بفتح الملف و الكتابه فيه و هذا يعنى أن أكثر من مستخدم يقوم بالكتابه فى نفس الوقت الى نفس الملف مما يتلف الملف و يجعله غير صالح للعمل و لحل هذه المشكله يجب أستخدام الدالة flock() التى تقوم بإغلاق الملف لأى محاولات أخرى من قبل مستخدمين أخرين للكتابه فى الملف أو تعديله .

و تقبل الدالة معاملين الأول هو مؤشر الملف المراد إغلاقه و الثانى هو عدد صحيح يعبر عن نوع الإغلاق و يتضح ذلك من الجدول الأتى :

الكود	النوع	الشرح
1	shared	يسمح الوصول المتعدد للملف لقراءته فقط و لا يسمح بالكتابه فيه و يستخدم فى حالة القراءة من ملف مشترك
2	exclusive	يمنع الوصول المتعدد للملف بغرض القراءة أو الكتابة و يستخدم فى حالة القراءة من الملف
3	release	يقوم بإلغاء القفل أو تحرير الملف من النوعين السابقين

ويجب عليك أن تقوم بتنفيذ الاغلاق مباشرة بعد فتحه و الغاؤه مباشرة قبل أغلاق الملف كما يتضح من الكود الاتى:

```
$fp = fopen( "myfile.txt", "a" );
flock( $fp, 2 ); // exclusive lock
// write to the file
flock( $fp, 1 ); // release the lock
fclose( $fp );
```

لاحظ أن عملية الاغلاق باستخدام الدالة flock() هو عمل احتراسى اى ان البرامج الاخرى التى لا تهتم باستخدام الدالة قد يسمح لها بالوصول الى نفس الملف فى نفس الوقت.

التعامل مع الأدلة الفرعية:

فيما يلى سنتعرف على الدوال التى تسمح لنا بالتعامل مع الادلة الفرعية بالقراءة او الكتابة او الحصول على معلومات عنها .

الدالة mkdir():

تقوم الدالة mkdir() بانشاء دليل فرعى و هى تقبل معاملين الأول هو مسار الدليل المراد أنشاؤه و الثانى هو حالة الدليل التى نريد نجعله بها و هو رقم ثمانى يتم كتابته دائما عن طريق صفر فى أول الرقم و هذا الرقم له معنى فقط فى النظام يونيكس و يتم عن طريقه تحديد صلاحيات الملف و يتكون من ثلاث مجموعات من الأرقام من 0 الى 7 تمثل الصلاحيات owner, group, everyone على الترتيب .

تقوم هذه الدالة بإرجاع القيمة true اذا تم انشاء الدليل الفرعى و القيمة false أذا حدثت مشكله فى الانشاء و عادة يتم ذلك أذا كان المطور ليس له صلاحيه لانشاء أدله على السيرفر .

```
mkdir( "mydir", 0777 ); // global read/write/execute
permissions
```

الدالة rmdir():

يمكنك أن تحذف دليل فرعى بالكامل أذا كنت تملك صلاحية لذلك و أذا كان الدليل الفرعى فارغ من الملفات و تقبل الدالة معامل واحد هو مسار الدليل المراد حذفه .

```
rmdir("mydir");
```

الدالة opendir():

لا يمكنك ان تقوم بقراءة محتويات دليل فرعى قبل الحصول على مؤشر لهذا الدليل و تقوم الدالة opendir() بذلك مهيئة الدليل للفتح و ترجع القيمة true أذا تم فتح الدليل بنجاح أو القيمة false أذا كان الدليل غير موجود مثلا :

```
$dh = opendir( "mydir" );
```

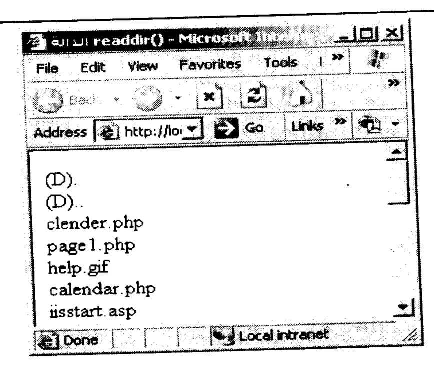
الدالة readdir():

تقوم الدالة readdir() بقراءة محتويات الدليل الفرعى من أدله و ملفات مثلها مثل الدالة fgets() و تتطلب معامل واحد هو مؤشر الدليل التى تقوم بقراءته

وتقوم بأرجاع الأسماء بدون المسار حتى نهاية الدليل التى تعطى عندها القيمة false .

مثال:

```
<html>
<head>
<title> الدالةreaddir()</title>
</head>
<body>
<?php
$dirname = "mydir";
$dh = opendir( $dirname );
while ( gettype( $file = readdir( $dh )) !=
boolean )
{
if ( is_dir( "$dirname/$file" ) )
print "(D)";
print "$file<br>";
}
closedir( $dh );
?>
</body>
</html>
```



لاحظ فى السطر رقم 10 أننا قمنا بأختبار نوع قيمة الدالة أذا كانت شرطية boolean و هذا لن يحدث الا أذا وصلت الدالة الى نهاية الدليل .

ثم نقوم فى السطر 12 بأختبار القيمة التى تم ارجاعها اذا كانت دليل يتم طباعة النص "(D)" بجانب أسم الدليل و بهذا نميزه عن الملفات الفرعيه .

لاحظ انه هناك طريقه أخرى للأختبار الموجود فى السطر 10 وهى :

```
while ( $file = readdir( $dh ) )
{
print "$file<BR>\n";
}
```

هذه الطريقه فعاله لانه عند عدم وجود قيم تستطيع الدالة readdir() قراءتها فأنها ترجع القيمة 0 لكن تحدث مشكله أذا كان هناك ملف أو دليل أسمه 0 فى

هذه الحالة فأن القيمة false يتم تحقيقها و يتم الخروج من الحلقة بدون أستكمال القراءة.

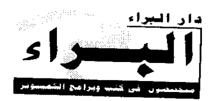
المحتويات

دَارُ البَــراء

بمصر وجميع الدول العربية

طبعة 2005

رقم الإيداع
2005/1898
ISBN
977-17-1956-4



المركز الرئيسى : 11 شارع د/محمد رأفت – محطة الرمل – الإسكندرية
تليفون وفاكس : 4838326 (03)(+2)
موبايل : 0101634294 (+2) - 0123357844 (+2)
Email : info@egypt***books***.net
URL: www.egypt***books***.net